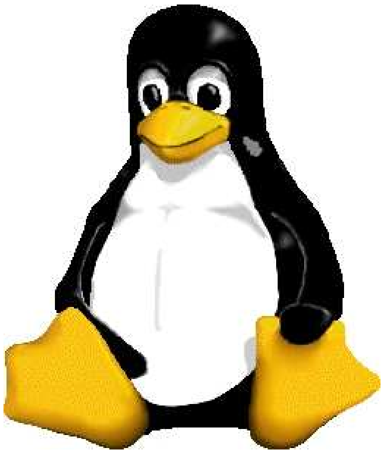
Instalación y Administración de Sistemas



Linux

Georg Lehner

19/08/2000

Índice General

1. Prefacio **7**

2. Introducción **9**

I. Manejo 11

3. Manejo de Linux en ambiente textual **13**
- 3.1. login 13
- 3.2. shell 13
- 3.3. línea de commandos 14
- 3.4. comandos básicos 15
- 3.5. manipulación de archivos 15
- 3.6. stdin/stdout/stderr 16

4. Manejo de Linux en ambiente gráfico **19**
- 4.1. Chooser y Login Widget 20
- 4.2. Gnome Desktop 21
- 4.3. Generalidades de GUI's 23
  - 4.3.1. Ratón 23
  - 4.3.2. Ventanas 24
  - 4.3.3. Gestores de Ventanas 24
  - 4.3.4. Interacción con el usuario 25

5. Scripts **27**
- 5.1. Gramatica para bash 28
- 5.2. Decisiones 29
- 5.3. Programas shell resumidos 30

II. Configuración 33

6. X-Windows **35**
- 6.1. Servidor y Clientes 36
  - 6.1.1. xinit, xstart. 37
  - 6.1.2. Gestor de Pantallas - xdm 38

6.2. Configuración 39
6.2.1. Categorías de tarjetas gráficas 39
6.2.2. Xconfigurator 39
6.2.3. XF86Setup 39
6.2.4. xf86config 39
6.2.5. XF86Config 39
6.2.6. Diagnósticos 39

7. Red TCP/IP **41**
7.1. Drivers 42
7.2. Protocolos 43
7.3. Tabla de enrutamiento 44
7.4. Servicios Internet 45
7.5. Configuraciónes ejemplares 46
7.5.1. Conexión telefónica a redes 46
7.5.2. Syncronización de una Portatil con una PC de Escritorio 55
7.5.3. Preparación del puerto paralelo para PLIP 56
7.5.4. Configuración del enlace TCP/IP 57
7.5.5. Observaciones adicionales 57
7.5.6. Intranet con LAN/Ethernet 58

8. Impresión **59**
8.1. LprNg e impresión remota 59
8.2. Cupsys 59
8.3. Ghostscript 59
8.4. Magicfilter 59
8.5. Redhat-Printtool 59

9. Redes Heterogeneas **61**
9.1. Redes Microsoft - Samba-Suite 61
9.1.1. Conectar Linux a un servidor SMB 61
9.1.2. Proveer Servicios SMB desde Linux 61
9.2. Redes Novell - ncp 61
9.3. Redes Appletalk netatalk 61

10. Usuarios **63**
10.1. Cuentas especiales, root, sysadmin 63
10.2. Cuentas normales 63
10.3. Grupos de trabajo 63
10.4. Authorización en la red 63
10.5. Como no ser root 63

11. Linuxconf **65**

III. Instalación **67**

12. Proceso de arranque **69**

13. Disquetes de arranque y rescate **71**

14. Disco duro **73**
14.1. Partitiones y sistemas de archivos 73
14.2. Lilo y grub 73

15. Instalación del Software **75**
15.1. Redhat 75
15.2. Debian 75
15.3. Perfiles, Paquetes, Fuentes 75

16. Hardware específico **77**
16.1. Núcleo modular 77
16.1.1. Objetos dinámicos 77
16.1.2. Módulos del núcleo 78
16.2. Recompilación del núcleo 82
16.2.1. Parametros de configuración 84
16.2.2. Versiones de núcleos 87
16.3. Instalación de un núcleo 87
16.3.1. Lilo 87
16.3.2. Módulos 87
16.3.3. Otros componentes 88
16.4. Unidad Zip en puerto paralelo 89
16.5. SCSI y escaneadores 89
16.6. Puertos en serie, modems y Plug and Play 89
16.7. Tarjetas de sonido 89
16.8. Adaptadores de Red 89
16.9. Niveles de ejecución y demonios 89

17. Programas **91**
17.1. StarOffice 91
17.2. Gnucash 91
17.3. Correo Electrónico 91
17.4. Fax, Voice 91

18. Bonboncitos **93**
18.1. Respaldos 93
18.2. Fips 93
18.3. TEX/LATEX, dvi y LYX 93
18.4. The Gimp 93
18.5. Linuxdoc 93

18.6. ps-tools 94

IV. Anexos **95**

A. Glosario **97**

B. Referencia rápida Emacs **99**

C. Referencia rápida vi **103**

D. Estándard de jerarquía de sistema de archivo **107**

E. Visualizadores **111**

1. Prefacio

Este manual se ha percebido como acompañante de los cursos sobre Linux que el autor ha dado en diferentes instituciones. Como tal no corresponde a un tema específico de computación, de Linux o de Hardware, sino se comprende como referencia rápida para problemas de uso comunes en torno al uso diario de una computadora, en este caso compatibles PC con el sistema GNU/Linux. Pretendo proveer los conocimientos necesarios para poder manejar, configurar e instalar un sistema Gnu/Linux a personas usuarias y administradores de computadoras en redes TCP/IP y/o MS-Windows, que usan, o planifican usar su computadoras en ámbitos profesionales. Para este fin el manual es diseñado también como introducción a los sistemas Unix y X-windows para facilitar la comprensión a quienes solamente conocen sistemas operativos diferentes. El contenido abarca temas como: manejo básico de un sistema Unix desde el shell, programación shell básica, manejo básico del sistema X-Windows, configuración de hardware, instalación y optimización del sistema, manejo de cuentas de usuarios, configuración y mantenimiento de redes TCP/IP y MS-Net bajo Linux, configuración de Modem y cuentas PPP, configuración del sistema de impresión remoto lpr, Ghostscript y utilidades.

Se requieren conocimientos avanzados de usuario de computadoras en ambiente texto y gráfico (Windows), y conocimiento de Inglés técnico en lectura.

La información presentada no es ni completa ni necesariamente es aplicable o correcta para cualquier sistema Linux, además del hecho, que GNU/Linux y sus distribuciones son un "blanco móvil", es decir presentan una evolución rápida y lo que hoy puede ser la solución estándard para un problema de administración de computadoras, mañana ha pasado a ser obsoleto – y así los manuales que describen estas soluciones.

Aprovecho este lugar para manifestar, que recibo con gusto sugerencias y correcciones, los cuales pueden dirigirse a mi dirección electróncia: `Jorge.Lehner@gmx.net`. Puede encontrar una versión html de este documento en Internet en la siguiente dirección: `http://www.ibw.com.ni/~toa`.

Quiero agradecer a dos personas, que no conozco personalmente, pero los considero muy especiales y que han hecho posible y necesario este manual, a través de crear un sistema operativo - Linux, y una nueva forma de considerar como compartir la propiedad intelectual y protegerla a la vez - Software Libre. Estas personas son Linus Torvalds, que no se contentó con los sistemas operativos ya existentes y tomo el valor de crear uno de nuevo, y Richard Stallman, que no se contento con las alternativas existentes - comprar Software o robarlo, y creó una alternativa, la de compartirlo. Sin embargo estos dos nombres solo son símbolos para miles más, que contribuyeron a hacer realidad estas dos visiones y mi principal agradecimiento es por vivir en el presente y presenciar la maravilla, que en un nuestro mundo considerado meramente egoista se manifiesta esta alternativa.

2. Introducción

El sistema operativo Linux ha comenzado como un proyecto de un aficionado. Sin embargo, en conjunto con la percepción de la licencia libre GPL y sus parecidos se ha convertido en un fenómeno. Dentro de un tiempo muy corto ha evolucionado, hasta convertirse en una verdadera alternativa de las soluciones más conocidas en el sector.

Este manual está dividido en tres partes. en la primera parte se encuentra una introducción básica para el manejo de Unix/Linux[1] en ambiente texto y en ambiente gráfico. Además cuenta con un pequeño instructivo sobre la programación del interpretador de comandos bash.

La segunda partes está dedicada a la configuración de los diferentes componentes de un sistema Unix/Linux. Primero se revisa con detalle la configuración del Servidor X Windows y las diferentes formas de su empleo y después la configuración de los servicios básicos: redes, conexión telefónica al Internet, impresión. El siguiente tema da una visión sobre el uso y la configuración de diferentes protocolos de redes ajenos a - pero disponibles en Linux y por último se tocan temas relacionados con la configuración de cuentas de usuarios.

La tercera parte se dedica a la instalación del sistema Linux, y a la configuración de Hardware específico y no soportado por la instalación básica. Así como la agregación y la deconfiguración de programas de aplicaciones.

En el anexo se encuentran materiales de referencia, que no solo apoyan en la lectura del texto, sino sirven durante el manejo del sistema. Referencia rápida de los editores Emacs y vi y otra información (hojala) útil.

[1] La mayoría de los temas no son específico para Linux, sino aplican a cualquier sistema Unix.

Parte I.

Manejo

3. Manejo de Linux en ambiente textual

3.1. login

En un sistema multiusuario (como Linux) tienen que diferenciarse los espacios (ambientes de trabajo) de los diferentes usuarios. Esto es una función del login que se encuentra como portal cada vez que se accede a la computadora. El segundo objetivo del login es la autentificación del usuario, o sea, detectar a través de una clave (secreta)=Password que la persona que utiliza el espacio de usuario indicado es este mismo. Es una medida básica de protección de los datos individuales de cada usuario.

Posibles formas de presentación son:

```
login: <nombre>
Password:
```

o

```
Username: <nombre>
Password:
```

Mientras el nombre del usuario es visible, la clave no se repite en la pantalla. Si, y solamente si, el par <*nombre*> <*clave*> coincide con un par de valores grabados en el sistema se provee acceso. En cualquier otro caso se denega el acceso. Es otra medida de seguridad que no se indica si fallo el *nombre* o la *clave*.

El *nombre* designa una cuenta (=account) que es el espacio o ambiente de trabajo del usuario.

3.2. shell

El sistema prove directamente después de la autentificación una facilidad para interactuar, es decir efectuar comandos y visualizar las respuestas correspondientes del sistema. Esta facilidad se llama procesador de linea de commandos = command line processor, o shell en la nomeclatura de unix. 'shell' es la cáscara de la concha y es un sinónimo para la capa visible desde exterior, encima del sistema operativo.

El shell lee una línea de texto que el usuario teclea, al apretar la tecla “Enter” interpreta el texto (por lo que también se le llama 'interpretador de línea de commandos=command line interpreter) y ejecuta los comandos encontrados. Si se cometió un error lo indica así. Al terminar los comandos el shell muestra una cadena de identificación llamado "prompt" y procede a leer la siguiente línea.

Existen diferentes shell, los tradicionales se llaman 'sh' y 'csh', otros son:bash, ash, zsh, ksh, csh, tcsh,... nosotros trabajamos unicamente con bash, que es compatible con sh y el shell official del software GNU.

3.3. línea de commandos

En una línea de commando se pueden concatenar varios commandos, separados por el símbolo ';' (semicolon).

Un commando tiene tres partes: nombre, opciones, parámetros. Solamente el nombre es obligatorio. Ejemplo:

```
ls -l --all *.c
```

nombre nombre del comando. Puede ser el nombre de cualquier archivo de programa. Ejemplo: 'ls'

opciones son indicaciones para el comando, que tiene que modificar su comportamiento. La forma tradicional es una sola letra y que es introducida con el símbolo '-' (guión). Ejemplo: -l
El proyecto GNU ha introducido en su implementación el uso de opciones explicativas (verbose) que son una palabra entera y esta introducida por dos guiones contiguos: '–'. Ejemplo: –all
Opciones por su parte pueden tener parámetros. Ejemplos: -s15, –with_gettext=yes

parámetros indican los objetos sobre cuales el comando debe actuar. Normalmente nombres de archivos o de directorios. Ejemplo: '*.c'

Tradicionalmente tienen que escribirse los parámetros antes de los parámetros, en los programas GNU muchas veces eso no está exigido, pero es buena práctica.

Wildcards o comodínes son símbolos que pueden aparecer en los nombres de archivos, a veces también en los de los directorios. El comodín es sustituido por el shell con todos los nombres de archivos que cumplen con su forma.

***** sustituye una cadena arbitraria de caracteres.

? sustituye cualquier caracter individual.

En el ejemplo de arriba se sustituye '*.c' con todos los archivos del directorio actual que terminan con los caracteres '.c'.

3.4. comandos básicos

Manipulación de directorios

ls

rmdir nombre remover un directorio

cdnombre “entrar” en un directorio; cambiar el directorio actual al indicado

du (diskusage) calcular el espacio consumido por todos los archivos de un determinado subdirectorio

df (disk free) calcular el espacio libre en un disco duro

Manipulación de archivos

cp

rm remover uno o varios archivos

find listar y manipular seleccionados archivos en un subdirectorio

locate El editor y visualizador primitivo

grepDescriptioncommando-?

commando–help muestra una descripción corta del comando.

mancommando muestra la página del manual de unix que describe el commando

infocommando hace lo mismo, pero con otro programa.

info por si mismo contiene un manual extensivo con hiperenlases, ejemplos y mucho más para todos los programas GNU.

3.5. manipulación de archivos

’cp’ puede copiar una **lista de archivos**, explícitamente nombrado, como directorio, o con wildcards hacia otro **directorio**, si el directorio no existe se crea, pero no se pueden especificar jerarquías de directorios como destino. Se puede copiar un archivo a otro, que tiene un nombre diferente. Si el destino existe se sobreescribe (!).

’ln’ crea enlaces. Esto no son archivos, sino referencias hacia archivos, que se pueden imaginar como entradas duplificadas en un directorio. Existen enlaces simbólicos (soft link) y directos (hard link). Los primeros refieren a la entrada original en el directorio, los segundos son copias directas. Por lo general se prefieren usar soft links.

Generalmente el comando ’mv’ se utiliza mas veces que el comando ’cp’. Tiene prácticamente la misma funciónalidad, menos que el archivo original se elimina. Por lo general ’mv’ es mucho más eficiente, ya que solo tiene que cambiar una entrada en el directorio, y no hacer copia del contenido del (posiblemente largo) archivo.

find

Itera por todos los archivos del directorio especificado (o actual), y repite esta función recursivamente por todos los subdirectorios. Este comportamiento se puede modificar, usando operadores (en forma de opciones con parámetros). Ejemplos:

find genera una lista de todos los archivos en el directorio actual y en sus subdirectorios

find

find/home-namecore-execrm{}\; encuentra todos los archivos de nombre core bajo el directorio /home y ejecuta el comando rm con el nombre correspondiente como parametro, es decir borra los archivos 'core'. La secuencia '{}' denota el nombre del archivo actualmente encontrado y la secuencia '\;' es necesario para concluir el parametro de '-exec'.

findrm{}\; parametros secuenciales se combina con una operación 'y' (AND). En este caso, todos los archivos core con una edad de mayor de siete días se eliminan. '-print' ademas imprime el nombre del archivo en la pantalla.

'find' es muy poderoso y se utiliza mucho en la administración automatizada de sistemas. Sin embargo es muy ineficiente ya que itera cada vez por todo el disco duro. Para encontrar la ubicación de un archivo se utiliza mejor el comando 'locate'. Este comando utiliza una base de datos indexados de todos los archivos en el disco duro, y extrae de esta la ubicación. Ejemplo: locate fernando; imprime todos los archivos que en alguna parte de su nombre contienen la cadena 'fernando'.

Si un archivo se renombra, se borra, se copia desde otro medio, etc. la base de datos no se actualiza y locate no reporta correctamente. el programa 'updatedb' actualiza (utilizando 'find') la base de datos y se correo automáticamente una vez a la semana.

3.6. stdin/stdout/stderr

Los comandos usados en los shell de unix pueden considerarse en la mayoría de los casos como 'filtros'. Ellos reciben una serie de caracteres (stream=corriente) por la entrada estándard (=stdin), los manipulan y remiten el resultado por la salida estándard (=stdout). Si hay que darle algunos mensajes al operador estos se transmiten por el canal de la salida estándard de error (=stderr). En el shell stdout y stderr por defecto son conectados y mensajes de procesamiento se intermezclan con la salida del programa.

La salida de un programa puede redirectionarse con el operador '>' hacia un archivo. La entrada puede obtenerse desde un archivo con el operador '<'.

La salida del stderr puede redirectionarse mediante el operador '&>' hacia un archivo, y de esta manera separarse de la salida estándard del filtro.

Con el operador '|' (pipe=tubo) se pueden concatenar dos programas, la salida del primero va entrando en el segundo programa. De esta manera se pueden crear cadenas de filtros, obteniendo efectos sofisticados con commandos sencillos.

cat recibe series de caracteres en stdin y los remite concatenados en la salida.

catamorfrancis.txt|grepfernando busca todas las líneas mencionados, pero solamente imprime las líneas que adicionalmente tienen la cadena 'fernando'

'grep' puede usarse con expresiones regulares y de esta manera ayudar a filtrar muy eficientemente archivos. Un uso básico de 'grep' es, encontrar un archivo con un contenido específico, si no se conoce el nombre.

sed

El nombre se deriva de 'stream editor'.

sed

El editor tradicional de los unices es 'vi' - sinónimo para "visual editor", es basado en 'ex', que manipula línea por línea. Sus comandos se encuentran en muchos otros programas filtros de unix.

El editor oficial del proyecto GNU es 'emacs'. Emacs no solo es un editor sino un ambiente de trabajo con capacidades y extension inmensos.

Otros editores son: jove - mini editor con semejancias a emacs, jed - de escala media, muy flexible, pico - el micro editor que viene con el sistema debian linux.

Estos editores trabajan en ambientes de texto, algunas también en ambiente gráfico.

La diferencia en relación con procesadores de palabras (Word, StarWriter, WordPerfect) es, que cada caracter de un archivo puede ser manipulado y lo escrito se graba en el archivo en formato ascii, caractér por caractér, mientras los procedadores de palabras utilizan "Formatos" de archivos propios, para agregarle al texto escrito información adicional sobre las fuentes de letras usados, tamaño, posición, gráficas, diseño de páginas etc.

Para la administración del sistema se necesita manipular archivos de solo texto = plain text, o ASCII - text (American Standard Code for Information Interchange).

4. Manejo de Linux en ambiente gráfico

Los sistemas operativos Unix han tenido la fama de forzar el/la usuairi@ a usar comandos crípticos desde la línea de comando. Mientras siempre queda abierta esta opción,

hoy en día no hay necesidad para un@ usuari@ común de temer el uso de una computadora con estos sistemas. El escritorio virtual se ha hecho un estándar en las computadoras Unix, y la mayoría de las aplicaciones profesionales y/más difundidos existen en versiones para diferentes variantes de Unix, permitiendo al/la usuari@ cambiar de sistema operativo e inclusive de computadora con la posibilidad de utilizar los mismos datos y la misma forma de trabajar en cualquiera de ellos[1].

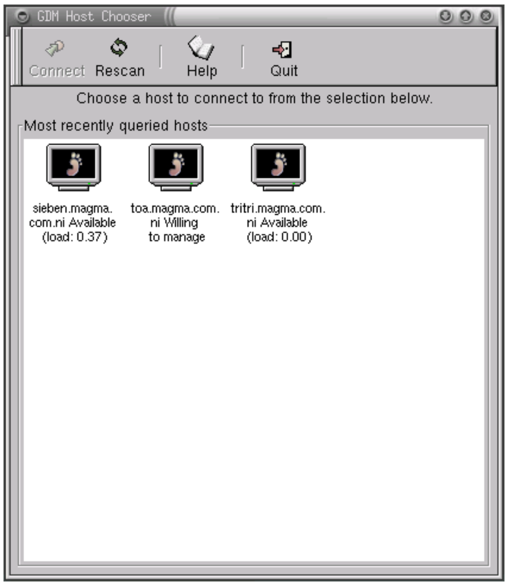
Esto fue posible en gran parte por una triada: la cuasi estandardización de Unix, la disponibilidad del servidor gráfico X-Windows, y el desarrollo de los sistemas de Escritorios virtuales - Desktop.

Los Escritorios virtuales han vivido una proceso de maduración desde su percepción, en donde el manejo y la forma de presentación se ha perfilado y su uso ha alcanzado una gran aceptación y asimilación. Con la creación de programas Desktop que se adhieren a

[1] Programas y sistemas elaborados de esta forma se llaman *interoperables*.

este perfíl se permite el manejo fácil de cualquier persona familiarizada con este estilo de trabajo, sin la necesidad de una capacitación específica a cerca del uso de la computadora. En este capítulo se presentará algunos elementos muy básicos del uso del escritorio Gnome Desktop, pero también ya se presenta el contenido en una estructuración que refleja la diferenciación de funciones de Interfáz, que hace los escritorios virtuales de Unix tan flexibles (y variados).

4.1. Chooser y Login Widget



En primer lugar hay que notar, que el ambiente gráfico en los sistemas Unix preservan las caracteristicas de seguridad que tiene el ambiente textual. Para acceder a una “sesión” gráfica hay que pasar por un proceso de autentificación al sistema. Esto se realiza a través de un Login Widget[2], que presenta un campo de entrada textual para el nombre de usuario, y otro para la clave de autentificación. Si el/la usuari@ se autentifica correctamente, el sistema permite la entrada a una sessión en la computadora.

X Windows, sin embargo es altamente integrado en el ambiente de red, de tal manera, que una sesión de usuario no necesariamente tiene que ejecutarse en la computadora local (donde está sentado el/la usuari@) sino puede efectuarse también en una computadora remota. Esto es parecido a una sesión telnet en el ambiente textual. Sin embargo necesitamos un mecanismo de conexión a esta computadora remota. En computadoras donde está habilitado el acceso a máquinas remotas aparece primero un selector de computadoras, llamado “choosee”, una ventana que presenta una lista de máquinas disponibles. El/la usuari@ selecciona con el ratón una de las computadoras (con doble-click), la cual a continuación presenta el Login Widget ofreciendo una sessión (remota) de la manera arriba descrita.

[2]Widget es “cosa” en inglés, en los ambientes gráficos de computación significa cualquier elemento gráfico de interacción con el usuario: ventanas, botones, etc.

Nota: Las dos gráficas presentadas corresponden a los Widget Chooser y Login del Gnome Desktop Versión Ximian. Hay numerosas diferentes representaciones de Chooser y Login y las versiones básicas proveidos con X-Windows, aunque son menos llamativos todavía tienen la mayor rapidéz y estabilidad.

4.2. Gnome Desktop

El "Desktop" o "Escritorio" es el conjunto de elementos de control e interacción y sus funciones que permiten a un usuario organizar el trabajo con programas y aplicaciones en un ambiente gráfico. Esto consiste en elementos de organisación de elementos visuales (Window Manager), manejo de machotes de organización (Session Manager), Elementos gráficos de control del sistema (Lanzadores, acceso a dispositivos, indicadores, visualizadores, etc.).

El proyecto Gnome implementa un Desktop completo, altamente flexible y configurable.

Existen otros Desktops, como p.e. el KDE-Desktop.

No es necesario trabajar con un Desktop "completo" para poder utilizar programas gráficos en X-Windows.

Los Elementos más importantes en el Gnome Desktop son:

Panel es una barra, ubicada generalmente en la parte inferior de la pantalla, donde se ubican iconos, llamados appletts, que activan funciones con "un solo click", o visualizan información del sistema o del estado del Desktop o de una aplicación. El elemento más importante es el "Menu Principal" o "Main Menu", una huella de pie en forma de 'G' frente a un trasfondo gris. Apretando este icono se desprende el menu que da acceso a casi toda la funcionalidad.

Otros iconos, llamados "lanzadores" o "launcher" pueden crearse libremente. Si se hace Click en ellos, lanzan (arrancan) el programa asociado con ellos.

Gmc Es un manejador de archivos, con el cual se pueden manipular dispositivos de almacenaje como discos duros, floppys, cdrom (montar, desmontar, etc.); directorios y archivos (borrar, mover, copiar, renombrar, visualizar). Gmc trabaja como browser (navegador) de archivos en una o mas ventanas propias. Puede manipular enlaces en forma de "URL", que le permite lanzar aplicaciones en dependencia del tipo de archivo, segun la asociación del tipo "MIME" del archivo. Enlaces en el Escritorio mismo se visualizan en forma de iconos en la pantalla para su inmediata manipulación con el ratón.

Control ▷ SectionX-Windows

X es una aplicación que nos permite utilizar programas con interfaz gráfico, quiere decir, que los comandos para operar los programas se pueden dar via movimientos del ratón, apretando las teclas del ratón, sin embargo también con el teclado. Los resultados obtenido de los programas se visualizan en fórma gráfica, p.e. con cambios de colores o movimientos de objetos gráficos en la pantalla.

X-Windows no interfiere en el proceso de interacción entre el usuario y los programas de aplicación, sino más bien provee un ambiente estandardizado para los programadores. X trabaja en forma Servidor - Cliente. En la computadora donde "estamos sentados" se ejecuta el "Servidor X", que recibe nuestros señales y los transmite a los programas de aplicación - los clientes. También recibe mensajes desde las aplicaciónes, las cuales convierte en las figuras gráficas correspondientes. De esta forma libera a los programas de aplicación de tener conocimiento sobre la forma de programar el adaptador gráfico de la computadora donde son utilizados y un programa escrito para X Windows puede funcionar en cualquier computadora o sistema operativo, siempre y cuando tiene acceso a un Servidor X. Esta forma de funcionamiento tiene otra ventaja, ya que no tienen que residir el programa de aplicación en la misma computadora como el Servidor X. Eso nos permite acceder a nuestro escritorio virtual remotamente, via una red local e inclusive desde el Internet.

Para el usuario la validéz de un sistema consiste en gran parte en la disponibildad de programas de aplicación. X Windows permite la ejecución de cualquier programa Unix textual a través de emuladores de terminales textuales. Estos son parecidos a la ventana DOS en MS-Windows, pero con la diferencia que este provee una simulación del ambiente de *ejecución* (no todos los programas funcionan), mientras un terminal texto en X-Windows provee al programa Unix una simulación del ambiente de *entrada/salida*, y la ejecución queda inerta.

Sin embargo, el uso de programas textuales se está reduciendo con la disponibilidad de una vasta gama de aplicaciones gráficas, especialmente con la venida de los Desktop KDE y Gnome, que se basan en poderosas y modernas herramienta de programación y permiten la inclusión del código original de programas textuales en ambientes gráficos.

4.3. Generalidades de GUI's

4.3.1. Ratón

Generalmente se observa en la pantalla un objeto pequeño, llamado cursor o puntero, que cambia su posición con el movimiento del ratón de la computadora. La forma y o el color del cursor pueden cambiár para dar retroalimentación al usuario. p.e. si es una flecha o una cruz se supone que el usuario está posicionando el cursor, dentro de áreas destinado para textos es una línea vertical en forma de 'I', si es una mano indica que se puede hacer una operación, muchas veces de cambio de contexto.

Apretando las teclas del ratón solo por un instante se llama "hacer Click", por el sonido que emite la tecla. Haciendo Click dos veces en rápida successión se llama "Doble Click"; tres veces: "Triple Click". Esto es opuesto a apretar el botón y no soltarlo y es interpretado diferentemente. Mantener apretado el botón mientras se mueve el cursor se llama "jalar", o "Drag". ¡Ojo! Generalmente se aprieta – mueve – suelta. Se dice "soltar" o "Drop" a la acción de finalmente soltar el botón después del movimiento.

Existen ratónes con solo un botón (MacIntosh), en PC's compatibles se utilizan generalmente ratones con dos botones, en X-windows es muy común el uso de ratones de tres botones. Existen ratones con una rueda de "navegación" y ratones con cuatro o más botones.

La cantidad de posibles combinaciones de Click, apretones y movimientos aumenta con la cantidad de botones, lo que permite usar el ratón como dispositivo de comando mas versátil, pero a la vez se necesita más aprendizaje y práctica para manejar un programa que utiliza combinaciones complejas de comandos de ratón.

En MS-windows el Click izquierda (botón izquierda) activa el elemento que se encuentra actualmente bajo el cursor - se dice que este elemento gana el foco "gain the focus", o: a partir de este momento los demas comandos, especialmente del teclado se dirigen hacia el elemento.

El doble Click izquierda "abre" el elemento, o sea activa el comando más frecuentemente asociado con el elemento.

El Click derecha abre un menu colgante del cursor, llamado "menu contextual", que prove una lista de los comandos que pueden operar en este.

En X-Windows estas reglas no son estandardizados, por lo que hay mucha variedad. Por lo general el Click y Doble Click izquierda tienen la misma función de "activar" el elemento, el Click o apretar derecha abre un menu contextual, y el apretar el botón mediano muchas veces prove funciones de movimiento del elemento selecionado.

4.3.2. Ventanas

Los elementos mostrados en la pantalla son figuras geométricas, generalmente rectángulos que representan “areas de control” de diferentes programas y sirven para la interactuación con el usuario de estos mismos.

En MS-Windows se llaman Ventanas, en X-Windows muchas veces se llaman “widgets”, que tiene significado de: útil, dispositivo de entrada.

X-Windows prove a los programas áreas rectangulares que pueden usarse para información gráfica y/o textual. La ubicación y el tamaño de este área es determinado por el “administrador de ventanas”, un programa que dibuja un “marco” alrededor del área, y permite al usuario mover, y redimensionar, esta ventana a través del mouse. También se puede “minimizar” o “iconificar” la ventana, reduciendola a un símbolo e inhibiendo interacción, hasta que se vuelva a “reestablecer” a su tamaño anterior, o se “maximisa”, lo que es darle a la ventana el tamaño máximo posible en la pantalla, tapando todos los demás objetos.

“Enrollar”, “shade” una ventana, es mantener su barra superior en lugar y tamaño, mientras se elimina graficamente el resto de su contenido. La operación contraria es “unshade”.

Mientras la maximización en MS-Windows hace actuar la ventana de una manera especial, p.e. no se puede cambiar su tamaño, en X-Windows muchas veces solamente es una operación que redimensiona el tamaño al máximo. Puede haber máximización horizontal (ancho), vertical (alto) y ambos.

Otra forma de modificación automática de tamaño es “fill” o llenado, donde se respetan ciertos otros elementos para no taparlos.

Para modificar el tamaño de una ventana interactivamente se “jala” de una de las esquinas, o de un borde de la ventana hacia la dimensión deseada.

También a través del marco y con comandos del ratón se puede terminar normalmente o esforzadamente el programa que “posee” la ventana, estos comandos a veces se llaman “destruir”, y “anihilar”, o “terminate” y “kill” respectivamente. En vez de destruir tambien se lee “delete”.

Pueden haber varias ventanas e inevitablemente uno se posiciona “encima” de otro cuando el espacio ya no alcanza para mostrarlas todas a la vez. Se establece un orden de “bottom” que es lo “mas a fondo” a “top” que es la ventana en primer plano que es completamente visible. “Raise” una ventana es subirla un “nivel”, “Lower” es bajarla. “Raise to top” es subirla a primer plano, mientras “send to bottom” es ponerla “abajo” en la “pila”.

4.3.3. Gestores de Ventanas

Para el manejo de las ventanas existe una gran variedad de formas y convenciones según el sistema operativo: McIntosh, NeXt-Step, MS-Windows, X-Windows “tradicional”.

Las últimas tendencias en la programación en X-windows proveen administradores (gestores) de ventana cada vez más sofisticados, y configurables.

Mientras hace poco la noción ha sido la de modificar un gestor de ventana existente de tal manera que provea un "look&feel" - "mirar y sentir", parecido a un determinado sistema, actualmente se diseñan nuevos gestores, que son capazes de emular diferentes sistemas desde un inicio.

En efecto disponemos de una gran variedad de gestores de ventanas, desde los tradicionales hasta los más sofisticados. A continuación una lista de gestores en aproximadamente este orden, que a la vez representa "velocidad de operación" decresciente:

twm Tab Window Manager. Es importante, porque "siempre" funciona y se usa como último recurso si otros gestores fallan o no son presentes. Pequeño y rápido. El botón izquierda en el desktrop abre un menu de programas, el botón deracha operaciónes del gestor y la salida de X, y el botón en el centro operaciones en ventanas (kill, move, etc.) – (¡Verificar esta información!).

mwm Motiv Window Manager. Motif es un ambiente de programación comercial muy popular. Una versión libre de Motif es LessTiff.

fvwm Free Virtual Window Manager. Es un succesor de twm. Utiliza menos memoria durante su corrida e introduce el concepto de escritorios virtuales que son conjuntos de ventanas. Cambiar a otro desktop significa visualizar en el monitor otro conjunto. Se pueden configurar muchos diferentes conjunto y sin salir de X trabajar así en varios diferentes "ambientes".

fvwm2/95 Modificaciones de fvwm para aparentarse a MS-Windows-95.

AfterStep Simula el Desktop NeXT-Step, extremadamente elegante. NeXT fue una generación de computadoras diseñados por una empresa derivada de MacIntosh.

ICEwm Gestor bonito y flexible con barra de tareas/sistemas.

Enlightenment Gestor principal del Gnome Desktop. Configurable con Scheme (Lisp).

WindowMaker Gestor preconfigurado de Debian Linux. Tiene una presentación y un manejo alternativo a los gestores parecidos con MS-Windows.

Sawmill También configurable con Scheme. Mi preferencia porque es más rápido que Enligthenment y muy versátil.

qvwm Gestor que simula Windows 95 y ocupa muy pocos recursos.

wm2 Un gestor definitivamente mínimo.

4.3.4. Interacción con el usuario

Follow\-toFocus hay que hacer click, o inclusive doble click para una ventana reciba el focus.

RaiseDescriptionManualDescriptionStackDescriptionrandom se escoje un lugar arbitrario para la nueva ventana.

firstDescriptioninteractive lo mismo como Manual Placement

Animate se agrega efectos gráficos, a los transformaciones de ventanas. p.e. al crear una ventana se deja aparecer de alguna orilla de la pantalla de donde se desplaza visiblemente hasta llegar a su destino final. O al minimizar y restablecer se simbolisa que la ventana se encoge y estira respectivamente.

5. Scripts

Con script (escrito) se denomina programas que se escriben en texto claro y que son interpretados directamente, o sea, que no tienen que pasar por un proceso de traducción. En otros ambientes se llaman también archivos batch. Los lenguajes script se distinguen en el sentido, que ellos mismos solo tienen la funcionalidad de aglutinar comandos "externos", los cuales pueden ser entonces de una complejidad arbitraria, sin que el lenguaje script mismo tenga muchos comandos. Existe un sinnúmero de lenguajes script, algunos ejemplos son Tcl, perl y la gran variedad de shell's (interpretadores de comandos) como son zsh, csh, ksh, sh. El shell estándard del proyecto GNU y de Linux es bash e incorpora un lenguaje script muy poderoso.

Podemos ver un script como una listas de comandos que el sistema linux permite ejecutar automáticamente, en vez de tener que introducirlos manualmente. Hay tres conceptos que se asocian con los lenguajes script, o la programación shell: la creación de "macros", o sea subsumir bajo un solo nuevo nombre de comando una seríe de commandos indivduales, la realización de tareas complejas a través de la combinación de varias tareas sencillas - divide and conquer, y la realización de tareas diferentes pero parecidas con un solo comando - parámetros.

Para crear un script se anotan los comandos individuales en secuencia en un archivo ascii, el cual se puede interpretar a continuación, utilizando el nombre del archivo como comando shell. Para esto se le necesita dar permiso de ejecución al archivo, ej. ("macro"):

Escriba en un archivo de nombre 'archivos' la línea: "ls", ejecute el commando: "chmod +x archivos", y a continuación "./archivos"

El comando 'ls' se ejecutara.

Vamos a ver este proceso más detallado:

1. Bash reconoce, que "./archivos" no es un comando interno.

2. Normalmente busca en la via de acceso (Variable PATH) si encuentra el archivo. En este caso, el nombre del archivo está especificado absolutamente ('./' es el directorio actual, y 'archivos' existe) entonces lo trata de ejecutar.

3. El atributo de ejecución está activado entonces se determina si el archivo es en uno de los formatos binarios que se pueden ejecutar directamente. En este caso es un archivo de texto, no 'binario'

4. Si la primera línea tiene la secuencia "#!" seguido por el nombre de un comando (filtro), este comando se ejecuta, usando el archivo script como entrada en stdin.

5. En nuestro caso no existe esta línea, por lo que se ejecuta el shell por omisión (default shell) como filtro.

6. Este nuevo bash (sub-shell) lee la línea 'ls', y lo interpreta como si hubieramos typeado el comando a mano: Se ejecuta 'ls' y los archivos del directorio actual se alistan en la pantalla.

7. El nuevo bash termina y regresamos al prompt del bash originador.

5.1. Gramatica para bash

inicia un comentario, el resto de la línea se ignora

$ inicia una sustitución de variable
seguido por un número 0 - 9 reproduce en su lugar el texto del 'token' 0 - 9 respectivamente en la línea de commando
seguido por un texto reproduce el contenido de una variable shell
seguido por una expresión reproduce el texto del stdout de la expresión, ejecutandola en un sub-shell.

**** preserva el siguiente carácter si este es especial, o introduce caracteres especiales, \\\\ se convierte en \\, se llama “escape caracter”.

\\n nueva línea (newline) ; '\\\\n' se convierte en '\\n'

\\r retorno al inicio de la línea

\\t tabulador

\\g suena el timbre del terminal

' (quote, single quote), protege una secuencia de caracteres y palabras del procesamiento de línea de bash; 'au\\nque no' se mantiene intacto, inclusive '\\n'

“ (double-quote) igual como el single quote, pero “\\” y “$” y ' preservan su función, “\\” solamente si es seguido por $, ‘, ", \\, or <newline>.

.StandardVariables bash, son nombres que pueden tener asignados cadenas de caracteres arbitrarios. Comunmente se utilizan letras mayusculas, para mas fácil diferenciarlos de los comandos.

Variables pueden ser leídos y escritos en cualquier momento. La preconfiguración de la computadora asigna un juego completo a diferentes variables tradicionalmente usados en unix. El juego de este variables permite detectar desde los programas y comandos en que estado y con que usuario etc. se encuentra el sistema en cualquier instanto. Porque definen el ambiente de ejecución las variables también se llaman “variables de ambiente” o “ambiente” (Environment variables, Environment).

El comando interno de bash 'set' escribe en stdout todas las variables de ambiente y su valor. Para definir una variable simplemente se le asigna un valor:

```
VARIABLE=”Vamos a la playa”
```

Las variables solamente son válidos para su ambiente. Si se ejecuta un sub-shell ya no son “visibles”. Ej: “echo $VARIABLE” no imprime nada en stdout. Para hacer válida una variable se tiene que “exportar”; después de definir la variable se da el comando: “export nombre” con el nombre de la variable. De esta manera se pueden crear jerarquías de validéz de variables (scope). Las variables que solo los usamos temporalmente son “protegidos” de modificaciones de sub-shells.

Bash ofrece una sintaxis simplificada: “export CAJA=523” es lo mismo como: “CAJA=523; export CAJA”, sin embargo se prefiere utilizar la versión original en archivos script, para no crear incompatibilidad con ’sh’.

5.2. Decisiones

La vida real nos exige tomarlas. Bash incluye comandos, que nos permiten

1. Bifurcar
2. Iterar

Estos dos elementos se conocen como control de secuencias. La bifurcación se escribe de la siguiente manera:

```
if TEST-COMMANDS; then
    CONSEQUENT-COMMANDS;
[elif MORE-TEST-COMMANDS; then
   MORE-CONSEQUENTS;]
[else
    ALTERNATE-CONSEQUENTS;]
fi
```

Más especial:

```
if [ algunos comandos ]; then
   lista de acciones
else
   otra lista
fi
```

El código de salida (exit status) del último comando de ’algunos comandos’ significa ’no válido’ si es 0, en todos otros casos significa ’válido’. Cada comando (inclusive los script) returna un valor numérico de salida al shell que ejecuta. En el caso más simple el shell retorna 0 si el comando se ha encontrado, otro valor en caso que no. Todos los comandos y las funciones shell describen minuciosamente como se genera su exit status, de donde se puede deducir como utilizarlos para la bifurcación. El exit status del últio comando se tiene disponible en texto mediante la variable especial $?.

La iteración se escribe de la siguiente manera:

```
for VAR [in WORDS ...];
   do COMMANDS;
done
```

donde VAR es una variable shell creada en el acto, que se asigna en torno a cada cadena de caracteres presente en WORDS. si se omita se sustituye por $@, los parametros posicionales.

5.3. Programas shell resumidos

Veamos algunos ejemplos reales[1], que nos pueden servir como machotes en caso de tener que resolver una tarea parecida:

Para leer archivos "HOWTO" comprimidos:

```
#!/bin/sh
if [ "$1" = "" ]; then
   ls /usr/doc/faq/howto | less
else
   gunzip -c /usr/doc/faq/howto/$1-HOWTO.gz | less
fi
```

Remover recursivamente archivos temporales y de respaldo de diferentes aplicaciones conocidas y comprimir ciertos tipos de archivos:

```
#!/bin/sh
#SQUEEZE removes unnecessary files and compresses .tex and README files
#By Barry tolnas, tolnas@sun1.engr.utk.edu
# echo squeezing $PWD
find $PWD \( -name \*~ -or -name \*.o -or -name \*.log -or -name \*\#\) -exec rm -f
find $PWD \( -name \*.tex -or -name \*README\* -or -name \*readme\* \) -exec gzip -
```

Convertir nombres de archivos a menusculas:

```
for i in * ; do [ -f $i ] && mv -i $i `echo $i | tr '[A-Z]' '[a-z]'`; done;
```

Un script que hace lo mismo:

```
#!/bin/sh
# lowerit
# convert all file names in the current directory to lower case
# only operates on plain files--does not change the name of directories
```

[1] Los ejemplos en parte no funcionan porque dependen de una versión Linux específica. El adaptarlo a su sistema es ejercicio para el o la lector(a).

```
# will ask for verification before overwriting an existing file
for x in `ls`
   do
   if [ ! -f $x ]; then
      continue
    fi
   lc=`echo $x | tr '[A-Z]' '[a-z]'`
   if [ $lc != $x ]; then
       mv -i $x $lc
   fi
   done
```

Remover archivos core:

```
#!/bin/sh
 USAGE=''$0 <directory> <message-file>''
 if [ $# != 2 ] ; then
     echo $USAGE exit
fi

echo Deleting...
find $1 -name core -atime 7 -print -exec rm {} \;

echo e-mailing
for name in `find $1 -name core -exec ls -l {} \; | cut -c16-24`
do
   echo $name cat $2 | mail $name
done
```

Parte II.

Configuración

6. X-Windows

En este capítulo se discutan conceptos detraś del sistema X-Windows, que lo hacen ser muy distinto de otros GUI (Graphical User Interface - Interfáz Gráfico de Usuario). Cabe mencionar aquí otros sistemas de interfáz de usuario gráfico en el ámbito de las computadoras personales: MS-Windows con todas sus variantes y derivaciones, MacOS de Apple Computers, Plan9, NeXT-Step (ya no se produce), y finalmente X-Windows, que por su larga historia y su versatilidad se encuentra en uso desde computadoras personales, hasta en Workstations y computadoras grandes, en su mayoría con sistemas operativos de la familia Unix.

X-Windows no es parte del sistema operativo, sino una aplicación, que establece una “norma” (API) para el acceso a recursos hardware gráficos - la tarjeta de video, y de entrada - teclado, ratón, joystick, grafic-pads. Consiste de esta manera de un “servidor” gráfico, al cual pueden acceder programas “clientes” para interactuar con las usuarias. X-Windows tiene una alta integración en el ambiente de red, por lo que el servidor y el cliente no tienen que estar ubicados en la misma computadora.

Las distribuciones de Linux incluyen el sistema XFree86, que es un derivado del original X-Windows, específicamente adaptado para computadoras de la plataforma x86. El interfáz gráfico ya viene preconfigurado para trabajar de una forma “workstation” o “personal computer”, o sea, su configuración se orienta en lo costumbrado de MS-Windows: una persona, una computadora. Estas preconfiguraciones son normalmente sencillos en su configuración, al menos en cuanto al Hardware, ya que hay una vasta cantidad de Chipsets de Video diferentes y XFree86 no desarrolla(ba) facilidades de instalar un driver (gestor) correcto. La adaptación de X-Windows a una tarjeta gráfica específica entonces puede resultar una tarea tediosa.

Quien está interesado solamente en una configuraciones Workstation estándar pero encuentra dificultades con XFree86 puede pasar directamente al capítulo de la configuración de Hardware (Tarjetas gráficas).

En los próximos subcapítulos se explicará la arquitectura del sistema X-Windows en más detalle para dar el fundamento a la configuración más avanzada de los diferentes componentes que integran el sistema operativo, la red, el interfáz gráfico y las aplicaciones en un sistema operativo Linux.

X-Windows fue percebido en el MIT - Massachusets Institute of Technologie, donde posteriormente los derechos fueron trasladados a X-Consortium Inc. y finalmente a la fundación de Software abierto: Open Software Foundation[1].

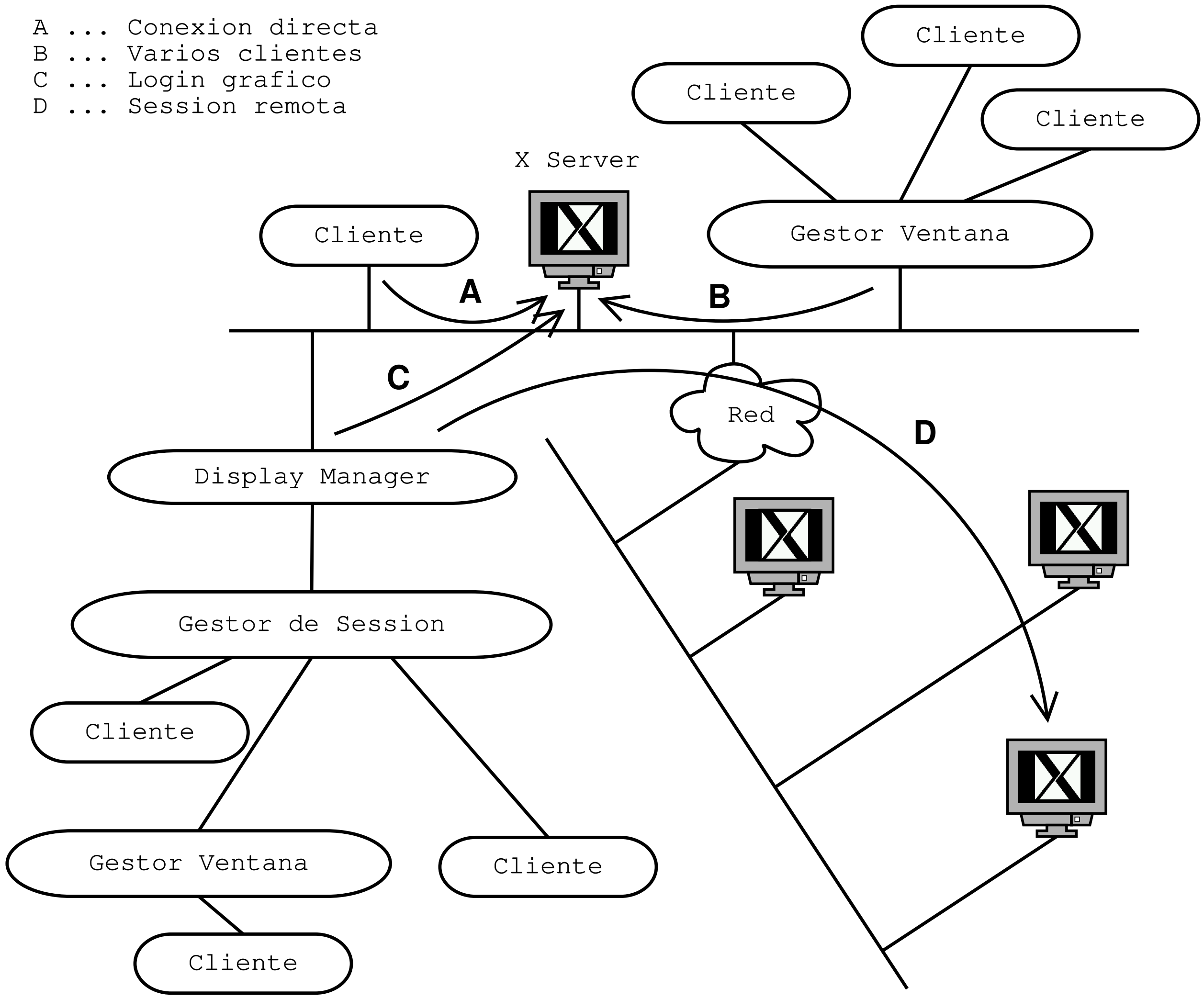
6.1. Servidor y Clientes

En el capítulo 4.2 se hace mención de la estructura cliente - servidor del sistema X-Windows. 'X' es el programa servidor, que "captura" los recursos pantalla, teclado, dispositivo puntero (y eventualmente otros) a la hora de arrancar, y permite a los programas de aplicación solamente el acceso a través de un protocolo específico con él. En cambió provee funciones avanzados para la graficación que pueden ser usados por los programas.

Para que un programa (cliente) pueda acceder a las funciones gráficas, tiene que *establecer una conexión* con el servidor X. Con un servidor pueden conectarse zero, uno o varios clientes; en efecto normalmente son una gran cantidad, ya que cada ventana representa un cliente.

La conexión puede realizarse de varias formas diferentes.

[1] No hay que confundir Software abierto con Software Libre. Ambos tienen percepciones muy diferentes y hubo mucho disputo sobre la liberación de X-Windows al público.



6.1.1. xinit, xstart.

Se puede utilizar solamente la "interior" parte de este esquema y correr un Servidor X. Para esto existen dos script: xinit y xstart. El segundo utiliza el primero para lanzar un servidor X local y algunos programas clientes, generalmente solo un gestor de ventanas o eventualmente de sesión. xstart provee un ambiente mas "seguro" y elaborado. La configuración de este modo de trabajo se efectua a través de los archivos xinitrc (global) y .xinitrc (individual por usuario).

Finalmente es posible correr solo una aplicación gráfica sin uso de un manejador de ventanas.

6.1.2. Gestor de Pantallas - xdm

xdm es el "X Display Manager", un programa que maneja conexiones entre clientes de X-Windows y Servidores a través de la red con un protocolo tcp/ip llamado xdmcp - "xdm - control protocol".

Vemos la parte de la conexión entre el servidor X y el xdm.

En la forma más sencilla, el servidor X corre en la misma máquina como el xdm. Normalmente entonces xdm lo lanza desde el archivo /etc/X11/xdm/Xservers. La comunicación entre los dos programas se establece a traves de un socket unix, o sea el núcleo del sistema y es muy eficiente. De esta manera, varias tarjetas de video, pueden ser manejadas por un xdm.

La segunda opción es para un X server solicitar directamente una conexión con un server a través de la opción de la línea de commando -query. Ej. "X -query toa.magma.com.ni". Si en la computadora toa.magma.com.ni corre un xdm y accepta la conexión desde la computadora solicitante entonces se establece la conexión a través de la red. El servidor X tiene que lanzarse entonces de otra manera, p.e. a través del archivo /etc/inittab. Lógicamente pueden residir los dos programas en la misma computadora.

La tercera opción es, que un servidor X solicita una conexión indirecta (X -indirect toa.magma.com.ni). De esta manera el xdm de contacto retransmite la solicitud de negociación a todos los xdm en la red, el servidor X por lo tanto se hace "público" en todas las computadoras conectadas (con xdm corriendo).

Vemos la parte de los programas clientes.

Estos solamente se pueden lanzar desde dentro de una "session". Para iniciar una sessión se requiere de un proceso de autentificación del usuario, en parte por la comprobación de derecho al acceso al recurso (del X server), pero en otra parte para preparar el ambiente de trabajo para el usuario, quiere decir las configuraciones individuales de todos los programas y aplicaciones.

El usuario que está sentado frente a un terminal en el cual corre un X server. Si el servidor usa conexión:

Indirecta: Si este trabaja con conexión indirecta xdm entonces presenta una lista de todas las computadoras que ofrecen negociación mediante xdm a través de un programa llamado "chooser". El usuario selecciona una computadora con la cual quiere trabajar. Después pasa a la misma situación como en la conexión:

Directo/local El xdm presenta una ventana de autentificación (login-widget) en la cual el usuario puede introducir nombre y clave. Tomese en cuenta, que estos textos pueden transmitirse a través de la red y de esta manera exponerse a un "intruso":

Después de la autentificación (exitosa) se lanzan programas definidos en /etc/X11/Xsession, que normalmente proveen un gestor de sessión, un gestor de ventanas y aplicaciones iniciales según configuración individual de cada usuario. Todos estos programas son clientes conectados con el servidor X a través de uno (local/directo) o dos (indirecto) servidores xdm. Cuando el último cliente hay terminado, lo que pasa normalmente al "cerrar" la sessión (el gestor de sessión) el servidor X se desconecta del servidor xdm y vuelve a conectarse con la solicitud directa o indirecta, o en caso de un servidor local va a ser lanzado de nuevo desde xdm.

6.2. Configuración

6.2.1. Categorías de tarjetas gráficas

6.2.2. Xconfigurator

6.2.3. XF86Setup

6.2.4. xf86config

6.2.5. XF86Config

6.2.6. Diagnósticos

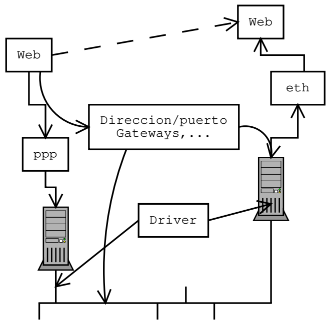
7. Red TCP/IP

El Internet hoy en día es una palabra que designa una red que físicamente abarca todo el planeta tierra, mas común aún se utiliza para describir una aplicación específica: www, o http. Pero en su percepción original es una colección de servicios y protocolos de red. TCP significa Transfer-Control-Protocol, e IP significa Internet Protocol. Si hablamos de redes Internet puede ser entonces la implementación mundial específica, pero también si en una oficina se conectan dos computadoras Linux via una red se trata de una red Internet. Para distinguir esta red pequeña de la red Internet mundial se le suele llamar Intranet a estas configuraciones, aunque el funcionamiento es idéntico.

La conexión de dos o mas computadoras siempre sigue los mismos mecanismos, los que se han identificados y normados por ejemplo en el modelo ISO-OSI de siete capas. Sin entrar en el detalle de este modelo nos adherimos al modelo de capas. Solamente contemplamos las capas de interés para la configuración del nucleo Linux. Desde "abajo" hacia "arriba" diferenciamos entre:

- Driver (gestor). Programa que controla el dispositivo físico de transmisión, tarjeta de red, modem, etc.
- Protocolo. Norma de intercambio de datos entre dos dispositivos. PPP, TCP/IP, PLIP, etc. programas que implementan esta norma bajo utilización de un Driver.
- Enrutamiento. El mecanismo que determina el camino que toman los datos a transmitir si existen varias conexiones (en Linux mínimamente dos si usamos cualquier conexión a una red), routing table.
- Servicios. Programas auxiliares utilizados en la administración de conexiones entre diferentes computadoras. DNS, ICMP, ARP, SMTP, etc.

El enrutamiento en realidad es parte de las capas inferiores. En la figura 7 se ilustra con la línea rayada la comunicación entre una computadora (izquierda) que utiliza el protocolo http para solicitar información de otra (derecha). Seguimos ahora el flujo de comunicación "real" desde la computadora izquierda a la computradora derecha. Se requiere de la seleción del destino (Dirección/puerto - información para el enrutamiento), los datos son empacados en un protocolo (ppp a la izquierda). Los driver del hardware, presuntamente en este caso para manejar un modem convierte los datos en señales analógicos que se transmiten a través de la línea telefónica. La "red" dibujado abajo puede contener multiples caminos para los datos, pero el enrutamiento asegura, que los



Ejemplo de una conexión en red

datos llegan a la computadora específica indicada en la derecha, en nuestro caso presuntamente ya convertido en señales de una red local Ethernet. La tarjeta ethernet convierte estos en paquetes de datos, el driver "eth" extrae de estos la información TCP/IP y la remite en "texto claro" a la aplicación destino.

Se puede apreciar, que todos los componentes/todas las capas tienen que concordar en su configuración para que pueda establecerse una comunicación. A continuación en este capítulo se discuta la configuración de cada "capa" y los posibles problemas emergentes.

7.1. Drivers

Los driver de hardware es la parte del sistema operativo que comunica el mundo exterior a través de dispositivos de entrada/salida con los programas. Cada tipo de dispositivos requiere un driver específico. Por suerte hay ciertas normas, que permiten ordenar los dispositivos en grupos y escribir driver's genéricos para un amplio rango de dispositivos reales.

Los dispositivos más comúnes para la conexión en red son:

modem establece conexión a través de una línea telefónica mediante modulación/demodulación de datos en señales de diferentes frecuencias. Normalmente un modem es conectado en un puerto en serie (/dev/ttyS*), o lo simula. WinModems (HSP) no pueden ser usados en Linux, por un lado por falta de información técnica, ya que es propietaria, por otro lado porque funcionan un concepto técnico que requiere limitar la funcionalidad de la CPU de la computadora, lo que no se considera conveniente en Linux. Los protocolos usados para la transmisión de datos son PPP (Point-to-Point-Protocol), y SLIP (Serial-Line-Internet-Protocoll) que es obsoleto.

tarjeta"estrella" con un cable individual por cada computadora, utilzando un concentrador (hup) para la interconexión. Hay diferentes velocidades de transmisión, siendo el cable coaxial mas lento, seguido por UTP/10MHz y recientemente UTP/100MHz.

Las tarjetas ethernet (NIC - Network Interface Card) existen para diferentes buses internos: ISA, PCI y otros. Según el circuito integrado que realiza la función de la codificación/decodificación hay diferentes "driver", los más conocidos son NE2000, DEC/Tulip, 3COM. El dispositivo es accesible en linux como /dev/eth*.

puerto paralelo Como una solución barata para conectar dos computadoras (point to point) se utiliza un cable especial que conecta en vez de una impresora otra computadora con el puerto paralelo. El interfaz es físicamente el del puerto paralelo, aunque hay variedades, el dispositivo es llamado /dev/plip*, ya que el puerto utiliza otra forma de control a que cuando se conecta una impresora.

El protocolo usado se llama PLIP (Parallel-Line-Internet-Protocoll). Existe el modo 0 y el modo 1 de transmisión. El modo 1 es más moderno y rápido y necesito otro tipo de cable y un puerto paralelo avanzado, mientras el modo 0 trabaja con cables "parallel Nullmodem" estándard y con cualquier tipo de computadora.

La configuración de estos dispositivos y redes se discutirá a través de ejemplos de aplicación concreta en otros capítulos.

7.2. Protocolos

La transferencia de información en TCP/IP se realiza a través de paquetes IP, que pueden ser de clase TCP o UDP.

Ambos disponen de un encabezado (sobre) en el cual el destino y el remitente son indicados en forma de números IP, que son representados como cuatro números decimales con un rango de 0 a 255 separados por un punto. Ej: 192.168.3.45

Para usos especiales se realizaron modificaciones del estándard original, ejemplos son:

ppp Point to Point protocol. Un protocolo que se especializa en la transmisión de paquetes TCP/IP a través de canales intermitentes de un solo transmisor a otro, como pasa con líneas telefónicas. En este caso no hay requerimiento de enrutamiento (solo hay una posible ruta) por lo que se aplica compresión del volumen transmitido aprovechando la redundancia de los encabezados IP. ppp prove autentificación y negociación de múltiples parámetros de la transmisión, es un protocolo simétrico peer to peer, o sea, cada estación es independiente y autónoma y en ambos direcciones puede iniciarse el enlace.

ppp puede usarse favorablemente también para conexiones experimentales entre dos computadoras, p.ej líneas de transmisión "caseras", equipos de infrarojo, ultrasonido, radio, etc.

slip Serial Line Internet Protocol. Es el predecesor de ppp e inferior en calidad. Sin embargo se utiliza a veces para conexiones virtuales, p.ej. existe un programa en Linux que marca una línea telefónica por demanda - diald, que utiliza slip como enlace de control.

plip es otro protocolo de punto a punto, que utiliza el puerto paralelo de una computadora para enviar y recibir datos. En este caso los paquetes no se transmiten en forma serial sino de cuatro en cuatro bits (nibbles) o en de ocho en ocho (bytes) según la capacidad del hardware. plip es la norma de desensamble/ensamble de los paquetes y las reglas de sincrónización - handshake entre las dos computadoras involucradas.

Los números IP en Internet son asignados por instituciones coordinadas (p.ej. `www.nic.ni`), para que ningunas dos computadoras tengan configurados el mismo número. Si configuramos un Intranet nosotros tenemos que cumplir esta tarea. Existen playas de números "privadas" que son reservados en Internet para la configuración de Intranets. Un rango que usamos en este libro para ilustrar nuestros ejemplos es: 192.168.0.0 - 192.168.255.255

Por ende vale mencionar, que los números IP no designan computadoras, sino dispositivos de red. Comunmente una computadora, especialmente si es de mesa, solo dispone de una conexión a la red, y por lo tanto es accesible solamente con un número IP. En este caso solemos identificar la computadora con su número IP.

Sin embargo imaginémonos la situación de una computadora de mesa con un modem conectado al Internet, a la cual conectamos una portatil via el puerto paralelo. En este caso la computadora de mesa dispone de dos números IP, una hacia el Internet que es asignada en el momento de conectarse via ppp y otra, que nosotros configuramos a la hora de instalar el interfaz plip hacia la computadora portatil.

7.3. Tabla de enrutamiento

En computadoras de mesa y muchas veces en servidores sencillos es suficiente despachar los paquetes de información de una manera fija - estática. Para esto, el núcleo de Linux dispone de una tabla de enrutamiento - routing table. La tabla de enrutamiento es una lista de especificaciones de rangos (playas) de direcciones IP, asociados con el dispositivo de red que se utilizará para despachar paquetes con destino en este rango. Eventualmente se puede configurar un "gateway" que es un número IP que designa una computadora capáz de remitir un paquete hacia otros segmentos de la red.

rango-ip → dispositivo

rango-ip → gateway →dispositivo

El comando "route" es utilizado para modificar y analizar la tabla de enrutamiento. Evocando este comando sin parámetro arroja la lista de rutas (routes) en la pantalla. "route add ..." agrega rutas y "route del ..." borra rutas de la tabla. La palabra "default" es una abreviación para una ruta por defecto que entra en función cuando no hay ninguna ruta establecida para una dirección IP específica.

En este momento vale mencionar, que el enrutamiento por número IP es un "enrutamiento externo", que permite establecer comunicación entre dos computadoras en una red ámplia. Hay también un "enrutamiento interno", que permite realizar una variedad de conexiónes lógicas entre estas dos computadoras. Esto se realiza a través de los "puertos" TCP. Prácticamente cada servicio Internet tiene asignado oficialmente un puerto, que no es nada más que un número de 16 Bit, que identifica un canal de comunicación, dentro de un conexión física establecida entre dos computadoras. Entre todos los paquetes intercambiados se filtran para cada aplicación los paquetes correspondientes a su canal a través del número del puerto. Para poder utilizar el mismo servicio más que una vez, al comienzo de una conexión entre las dos computadoras se negocia en el puerto estándar un canal (puerto) libre, el cual se utiliza a continuación.

7.4. Servicios Internet

En el archivo /etc/services se relacionan nombre de servicios con su respectivo número de puerto. Este archivo es coordinado por el comité regulador de Internet y viene con asignaciones estándardizados, sin embargo puede ser ampliado o modificado al gusto del administrador del sistema. Casi todos los programas clientes y servidores tienen la opción de especificar puertos alternativos a los indicados en /etc/services, sin embargo esta opción se usará solamente en casos especiales.

Todos los servicios implementan una relación cliente servidor, es decir, al conectarse a una computadora (remota) mediante un programa 'cliente' se establece una comunicación con un programa 'servidor'. Al establecer una conexión el servidor en muchos casos le exige autentificación al cliente y le puede denegar el servicio según el resultado.El servidor en muchos casos implementa un "lenguaje" de comandos en texto claro (ASCII). Estos lenguajes son definidos en los documentos RFC – Request-For-Comment lo que en realidad significa propuesta para comentarios, sin embargo suelen a llegar a ser "normas".

He aquí algunos (pocos) de los servicios de red que se usan más frecuentemente.

telnet 'sniffer' en texto claro. Inclusive la clave del login se transmite sin encripción.

Telnet se puede utilizar para comprobar la disponibilidad de un servicio IP en un host mediante la especificación del puerto correspondiente.

ftp es un protocolo para la transferencia de archivos. Como tál permite el acceso al directorio home del usuario autentificado y puede transmitir archivos del y hacia la computadora remota (get archivo, put archivo). con mget y mput se pueden seleccionar archivos múltiples, con dir o ls se puede leer el contenido de un directorio. Ftp transmite en modo binario (image) o en modo ascii, hay que cerciorse siempre que se utiliza el modo binario, especialmente el cliente MS no lo tiene configurado por omisión.

Eventualmente (y en Internet muy común) se permite acceso a un usuario 'anonymous' al cual se le pide como clave su dirección de correo electrónico. A este usuario se le brinde acceso 'world' a un directorio especifico, donde se depositan archivos disponibles para la publicidad. ftp-anonymous crea una debilidad de seguridad en un servidor.

http es un protocolo para la transferencia asimétrica de archivos. La transferencia es solamente desde el servidor hacia el cliente. Este servicio se utiliza para la carga de páginas Web. Las solicitudes de los archivos se efectuan a través del envió de un 'URL'-unified resource locator. Los archivos se llaman recursos, ya que pueden llevar información de cualquier tipo. El protocolo es mejor optimizado que ftp y no necesariamente se interrumpe la comunicación al ocurrir un error de transmisión.

smtp es el protocolo para el envío de correo electrónico. Smtp es un protocolo en texto claro y dado su carácter de transmisión de información confidencial el protocolo más problemático en cuanto a la privacidad. Esta situación ha llevado al desarrollo de técnicas muy maduras de encripción PGP y sus succesores GPG y otros. Smtp no solamente se utiliza para intercambio de mensajes entre humanos sino también para el acoplamiento de procesos técnicos y control de flujo. Smtp no provee un mecanismo de autentificación, ya que el enrutamiento del mensaje no se efectúa directamente sino a través de una meta-red de Servidores MX (Mailexchange). Un mensaje es injectado a la red en un servidor normalmente mediante un programa MUA (Mail-User-Agent) hacia un programa MTA (Mail Transfer Agent), el cual se preocupa por enrutarlo hacia su destino. En algún momento el mensaje llega a un servidor (MTA) que lo reconoce como mensaje local y lo deposita en su correspondiente apartado. Desde allí puede retirarse, otra vez con un MUA, o eventualmente con el protocolo pop (post office protocol) o imap. Entonces se reconocen tres fases de transferencia: MUA->MTA, MTA->MTA, MTA->Depósito (delivery).

pop3 el Post-Office-Protocol versión 3, se utiliza para recoger remotamente, de un casillero electrónico los mensajes. La mayoría de los programas lectores de correo (MUA) utilizan este protocolo para retirar correo del servidor. También es un protocolo que prove poca proteción de privacidad de los mensajes.

rpc Remote-Process-control es un servicio para ejecución remota de procesos. Fue diseñado junto con una variedad de otros protocolos por Sun Microsystems, como el nfs e yp. Estos protocolos utilizan un sistema de redirección de puertos, lo que implica un tratamiento diferente a los otros protocolos mencionados. La seguridad e integridad de estos protocolos ya no concuerda con los estándardes de la investigación actual y se han diseñado varias alternativas, sin embargo son tán populares – ya que proveen servicios que son altamente solicitado en trabajo en grupo y computación distribuida – que hasta hoy no han sido reemplazados.

nfs Network-File-System es el protocolo que permite compartir directorios de una computadora con otra a través de una conexión TCP/IP. Ambas computadoras tienen acceso a los archivos compartidos y para ambos se presentan como archivos común y corrientes de Unix.

otros tftp, dhcp, bootp, ssh, xdmcp, rsync, cvs, dns, imap, nfs, yp, etc.

7.5. Configuraciónes ejemplares

7.5.1. Conexión telefónica a redes

Este tipo de conexión es altamente vulnerable, ya que se basa en una técnica de transmisión y conexión de baja calidad y robustéz en comparación con otras tecnologías. Además hay varios niveles en los cuales pueden ocurrir problemas. Por eso vamos a analizar este ejemplo muy a detalle, aunque en la actualidad existen herramientas de configuración muy versátiles que simplifican sustancialmente la tarea de configuración y conexión.

A nivel de dispositivo podemos contar con un modem externo o interno.

Modem externo

Este se conecta a través de un puerto en serie, o COM-Port. Hay que configurar la velocidad de conexión entre modem y computadora, así como los demás parámetros, mediante el comando setserial. Mientras en viejas distribuciones se utilizaron puertos /dev/cua* actualmente se considera el uso de /dev/ttyS* donde ttyS0 = COM1, ttyS1=COM2, etc.

Modem modernos pueden utilizar hasta la velocidad máxima (IBM-Compatibles 115200 Bd), pero hay modem que requieren una velocidad específica máxima, un valor seguro puede ser 9600 Bd, muy seguro: 2400 Bd. Esta velocidad solo tiene que ver con la velocidad de conexión telefónica en el sentido, que no se puede transmitir más rápido en la línea telefónica que en la conexión entre computadora y modem. Sin embargo puede haber una conexión mas lenta en el teléfono que en el cable serial, así que es recomendable usar velocidades reducidas solamente para pruebas.

Hay que asegurarse, que se utiliza un cable correcto. Las computadoras PC utilizan tradicionalmente un conector DB25 para el puerto paralelo y para el puerto serial. Estos se distinguen, en que el puerto serial usa pines (macho). Con el modelo PS2 se introdujo el uso de conectores de DB9 (9 pines) para el puerto en serie, que por cierto es contra la norma RS232C, pero viene siendo práctico para reducir el tamaño de las cajas de computadoras. En todo caso el cable debe tener en un lado un conector DB25 con pines que se conecta en el modem en un conector "hembra", y entonces queda por lo general solamente la elección entre uno o dos conectores seriales en la computadora. Eventualmente se requiere de un convertido DB9 a DB25 - este también existe en dos formas y hay que fijarse bien cual tipo es necesario. También recuerda conectar el cable al modem primero.

Hay cables "incompletos" que no tienen conectados todas las líneas. Se puede realizar una conexión serial entre dos modem (o computadoras) con tres hilos, sin embargo esto no es recomendado porque requiere un mecanismo llamado Software-Handshake (Xon-Xoff) que no es tan confiable y además reduce la velocidad de transmisión de datos. En efecto en el conector DB25 no se ocupan todos los pines, en el conector DB9 sí. El uso de Software-Handshake requiere una configuración especial y no lo contemplamos aquí. Si utilizamos un cable incompleto (o con algún hilo quebrado!) pueden haber problemas de conexión, que no son detectables instantaneamente.

En caso que no hay posibilidad de distinguir cual conector corresponde a cual puerto serial hay varios posibilidades. Lo más recomendable es abrir la caja y verificarlo físicamente. A veces en la tarjeta madre o en la tarjeta serial se puede leer el número del puerto asociado con el conector respectivo.

Un método seguro es la utilización de un probador de puertos seriales en conjunto con un software para controlar el puerto, p.ej. el emulador de terminales "minicom", kermit, sayon, u otro.

Habiendo detectado el conector no necesariamente identifica el dispositivo ttyS* ya que este puede cambiarse eventualmente mediante jumper o en el CMOS-Setup (BIOS) de la computadora, así que posblemente hay que referirse al manual de la tarjeta y/o al CMOS-Setup de la computadora. Los valores estándard para los puertos utilizados son:

COM*	ttyS*	IO-Port	Irq
Com1	ttyS0	0x3f8	4
Com2	ttyS1	0x2f8	3
Com3	ttyS2	0x378	?
Com4	ttyS3	0x278	?

En las computadoras modernas puede que el puerto USB o el puerto infrarojo utiliza un dispositivo serial (el secundario), así que en el CMOS-Setup se puede configurar a cual puerto correspondiente al dispositivo restante (el primario), y con eso determinar el ttyS* a usar para el modem.

Modem interno

En este tipo de modems se ahorran los problemas con los cables, sin embargo es requerida la configuración del puerto y de la interrupción. Esto se realiza en modems “viejos” con conector ISA o EISA mediante el sistema PnP (Plug ’n Praise). La utilidad “pnpdump” arroja en la pantalla todas las posibles configuraciones para un modem PnP, en una forma que permite de manera fácil crear un archivo de configuración (p.ej. /etc/isapnp.conf). Este debe cargarse antes de usar setserial mediante la utilidad “isapnp”, (p.ej: isapnp /etc/isapnp.conf)

A veces se requiere reinicializar la computadora para este paso, inclusive puede ser necesario apagarla. No culpen al Linux por eso, hablen con ...

Modems internos con conector PCI también utilizan PnP pero “de lo bueno” y su configuración se puede detectar con la utilidad “lspci”, o con el comando “cat /proc/pci” (en núcleos con versiones inferiores de 2.4.0).

Generalmente se utiliza la velocidad de 115200 Bd para modems internos, ya que es un valor no relevante, pués no existe un conexión serial entre la tarjeta-modem y la computadora; sin embargo se tiene que usar un valor compatible con el driver serial.

Conectividad con el modem

Una utilidad bastante sencilla para comprobar la funcionalidad básica de un modem es el programa minicom, que permite conectarse al modem en modo “terminal”. Cada tecla que se aprieta se envia mediante el puerto serial al modem, cada caracter que el modem remite al puerto serial se visualiza como caracter en la pantalla.

Como root tiene que configurarse el puerto a usar (C-A O), es importante salir (C-A X) y volver a entrar de minicom para que los cambios tengan efecto (al menos conmigo). Al volver a entrar normalmente ya se puede apreciar un diálogo entre minicom y el modem. La mayoría de los modem utilizan comandos “Hayes” para su control. Estos inician con los caracteres “AT” (ATtention). El modem refleja cada caracter y remite una respuesta, si el comando es exitoso la respuesta es "CR-LF", o sea, el cursor de la terminal pasa a la siguiente línea.

A continuación una lista de comandos básicos:

AT*	Parámetros	Significado	Comentarios
ATZ		Reinicializar Modem	
ATH	1/0	"levantar"/colgar el telefono	A veces solo se puede usarATH y ATH0
ATD	# telefónico	Marcar un número (dial)	El modem espera estable-ceruna conexión de datos. Estose indica con "CONNECT".En caso que no se puede conectardespués de un tiempo se recibeun mensaje de error.
ATA		Contestar (Answer)	Si entra una llamada el modemenvia el texto "RING" altermnial. Si se contesta el modemespera sincronizarse con unmodem remoto y señaliza estehecho con "CONNECT".

El modem trabaja en uno de dos modos: en modo de "comando" podemos enviar los comandos al modem, pero después de establecer una conexión el modem trabaja en modo "transparente", donde todos los caracteres se envian al modem remoto es decir a la computadora remota respectivamente ya que el modem remoto también está en modo transparente. De este modo se puede regresar al modo comando por falla de conexión: interferenica, interrupción de la línea etc., o con una secuencia específica de caracteres.

Establecer un enlace

El primer paso para conectarse a un servidor internet es marcar el número, lo que efectuamos con el comando "ATD" seguido por el número del servidor. El modem marca primero el número y si la línea no está occupada se puede esperar que al otro lado contesta el modem del servidor. En este momento los dos modem intentan sincronizar la velocidad y el protocolo de transmisión sobre la línea telefónica, un proceso que no siempre es exitoso, especialmente con altas velocidades de transmisión y cuando la línea telefónica sufre interferencias.

Una vez que se "entienden" los modems lo notifican a la computadora; normalmente con el texto "CONNECT", y a veces seguido por la velocidad y los parámetros de la conexión, y pasan inmediatamente al modo transparente.

Ahora le toca el turno al proveedor del servicio. En la actualidad prácticamente todos los proveedores de servicios Internet via la línea telefónica utilizan el protocolo PPP para establecer el enlace de red, y nosotros vamos a enfocar en este.

Aunque PPP incluye un método de autentificación, muchos proveedores requieren una autentificación en texto pleno, normalmente con un diálogo como lo escribimos en los capítulos iniciales (3.1). Podemos realizar la autentificación manualmente, pero también hay utilidades para automatizar este proceso, dos de ellos (los más usados) son "chat" y "runscript". Este último es parte de minicom. Para curios@s refiero a las páginas manuales de estos programas y a las noches interminables que pueden pasar cuando empiezan a experimentar con ellos. Sin embargo hay que mencionar una cosa: La autentificación en texto pleno es un atentado contra la seguridad y es preferible cambiar de proveedor antes de acceptar esta clase de anacronismo. La utilización de scripts de conexión requiere por lo general escribir la clave de una (o varias) cuentas en un archivo texto y por lo tanto directamente legible para quien tenga acceso al archivo. Por lo tanto aumentamos también con esto la vulnerabilidad de nuestro sistema.

En algunos casos el proveedor configuró el servidor de tal manera, que después de la autentificación se lanza inmediatamente el servidor ppp, en otros casos nosotros tenemos que iniciarlo desde una línea de comando remota, por ejemplo con el comando "ppp<Enter>". En ambos casos hay que lanzar a continuación inmediatamente el enlaze ppp localmente. Si nos estamos conectando con minicom u otro emulador de terminal tenemos que terminar este sin reinizialización del modem o corte del enlace. Terminar: porque el puerto serial va siendo bloqueando por cada aplicación que lo occupa, para que no haya conflictos en su uso; Sin reinizializar: normalmente los programas que usan un puerto serial modifican sus parametros al usarlo y tratan de reestablecer los parametros anteriores antes de ceder el dispositivo. En el caso de minicom se puede lograr estas dos condiciones con la secuencia de teclas: "C-a q".

Dispositivo virtual de red: ppp

La conexión transparente entre las dos computadoras ahora se utiliza para realizar una conexión TCP/IP, mediante dos dispositivos de red, que se crean transitoriamente mediante el daemon pppd (en caso de una máquina Unix). Este es un programa que lee caracteres en su stdin y escribe en el stdout. Si lo lanzamos, especificando como uno de los parámetros de la línea de comandos un dispositivo de caracteres, pppd utiliza este dispositivo como stdin/stdout, en nuestro caso le especificamos entonces el puerto serial donde está conectado el modem.

pppd realiza la conexión con el pppd remoto en tres etapas:

1. Establecimiento del enlace via LCP (Link Control Protocol) y configuración de diferentes parametros de la red como MTU, número IP de ambos participantes, servidor de nombres, modos de compresión de datos/encabezados, etc.

2. Autentificación mutua. ppp es un protocolo simétrico y en principio ambas computadoras tienen que comprobar el derecho de conectarse a la respectiva otra parte. En el caso de un enlace hacia un proveedor de servicios internet normalmente el "cliente" no puede exigir autentificación al "servidor", un hecho que tiene que reflejarse en la línea de comando del pppd o en los archivos de configuración.

3. Intercambio de paquetes TCP/IP.

pppd puede ser configurado de una manera que le permite asignar al lado opuesto uno de un rango de números IP (eventualmente el rango solo consiste de un número), esto se hace cuando pppd es usado como "servidor". En "nuestro" lado local podemos acceptar la configuración propuesta remotamente, pero también se podría insistir en un número IP fijo. Otro parámetro importánte para el daemon local es, "defaultroute" que inserta en la tabla de enrutamiento de paquetes IP una salida por defecto a través del dispositivo transitorio que ha creado el daemon (ppp0, ppp1, etc.). Al cerrar la conexión la ruta por defecto original se restablece. De nuevo: si utilizamos ppp para conexiones, por ejemplo via un enlace radio hacia un subdominio de nuestra red comercial intranet, no se establecería una ruta por defecto, sino solamente para este subdominio, por lo cual la configuración sería diferente.

Existen dos formas de autentificación: PAP y CHAP.

Password Autentication Protocol el daemon local envia un nombre y una clave hacia el daemon remoto. Este por su parte compara los dos datos con todas las líneas del archivo /etc/ppp/pap-secrets (las claves en ppp son llamados secretos). Se se encuentra una línea correspondiente se autoriza la conexión, de otra manera el enlace se cierra.

La clave puede archivarse en /etc/ppp/pap-secrets en forma encriptada (unix-crypt) y se puede exigir que solamente secretos encriptados sean acceptados (en la configuración del daemon remoto - pruebelo con su proveedor). Crypt hoy en día no es una encripción fiable, cualquier computadora común puede descubrir un secreto encriptado con unix-crypt dentro de minutos.

Challenge Handshake Autentication Protocol Con este método nunca se intercambia el secreto en línea. El daemon remoto inicializa la autentificación, enviando su "nombre" y un número al azar - el "reto" (challenge). El daemon local produce un código único[1] desde el reto y desde la clave que encuentra en el archivo /etc/ppp/chap-secrets en la línea que contiene el nombre del daemon remoto. Este código único es remitida al daemon remoto, quien es el único que puede comprobar la validez, porque solamente él comparte el "secreto" y puede crear el mismo código único para la validación de la identidad del daemon local. CHAP usualmente repite el proceso de autenticación constantemente en ciertos intervalos. De esta manera se intenta evitar un ataque man-in-the-middle.

A continuación una línea de comando típica, que puede usarse experimentalmente para conectarse con un daemon ppp remoto:

pppd /dev/ttyS0 modem crtscts defaultroute noauth

Para averiguar problemas en una conexión se pueden utilizar las siguientes opciones:

[1] Pués, casi único. En realidad se produce un hash, o como dicen los matemáticos españoles: una función de dispersión.

debug escribe información sobre todos los paquetes de control en el syslog. Esta información puede rastrearse entonces con: “tail -f /var/log/syslog” (Debian), o “tail -f /var/log/messages” (RedHat).

logfile nombre escribe la información de purificación también al archivo “nombre”.

show-password muestra la clave en texto pleno en el archivo log. Redunda decir, que esta opción preferiblemente no se utiliza, mucho menos de manera constante.

kdebug # muestra en el archivo log también información de paquetes IP desde el núcleo.

Los archivos que contienen los secretos para PAP y para CHAP tienen la misma estructura:

```
# Comentario
# Nombre_de_cuenta Nombre_de_Computadora Secreto
l2310ka            ikarus          el_sol_quema
lodemas            *               3i4fe0es65
# un asterisco es válido para cualquier nombre de computadora
# en todas las conexiones menos con (ikarus) se usara el nombre lodemas
# con su secreto (encriptado)
```

Por fín hay que saber, que por lo general pppd es ejecutado sin opciones de la línea de comando, ya que todas las opciones se pueden grabar en los archivos de configuración, en primero lugar en /etc/ppp/options.

El mundo real es más dulce

En muchos casos desde una computadora no solo se conecta a un proveedor, sino eventualmente a varios, o se utiliza por lo menos diferentes números telefónicos para el enlace. Sea así, o haya solamente una conexión, no es factible que todo el mundo aprenda utilizar minicom, syslog y pppd desde la línea de comando para poder configurar su conexión telefónica a redes. A continuación una lista de programas que se han hecho para facilitar la configuración y la conexión, pero primero algunos aspectos desde el punto de vista de su utilización.

Se requiere de la *configuración* que se pueda introducir las características para una conexión. Estas se dividen en dos grupos: el dispositivo a usar: puerto serial, velocidad, modem, y los parámetros de conexión: número telefónico del servidor, nombre de la cuenta, secreto, y parámetros: autentificación previa, CHAP/PAP entre otros. Tienen que poderse manejar diferentes configuraciones, que se identifican y manejan a través del “nombre de la conexión”.

Después se necesita utilidades que facilitan el *control* de la conexión: establecerla, cortarla y poder saber si se está conectado o no, asi como el tiempo transcurrido, la cantidad de información enviada y recibida, y eventualmente el costo incurrido:

modemtool configuración: de RedHat, con un interfáz gráfico/X que es muy intuitivo en el uso.

wvdial nombre configuración: (elaborado por World Vision - wv) que detecta y configura el/los modem/s utilizado/s automáticamente y crea archivos de configuración llamado "nombre", que puede completarse manualemente con un editor de texto para agregar los datos de autentificación.

pppconfig configuración: la utilidad que prove Debian, cuenta con un interfáz gráfico/texto. Fácil de usar. Hay una conexón por defecto llamado "provider"

pon, poff control: son scripts que establecen/cortan la conexión con "provider". Se puede especificar otro nombre de conexión en la línea de comando.

masqdialer es un daemon en forma cliente/servidor, que permite el control de un modem para la conexión telefónica desde una red local. El servidor: mserver tiene que configurarse en una computadora donde ya existen todas las conexiones con su nombre. Estas (y otros datos mas) se le comunica a mserver a través de archivos de configuración. Un "dial"-cliente manda comandos al mserver via TCP/IP, y el mserver le prove información estatística sobre la conexión (tiempo, volumen, velocidad). Hay clientes texto, gráficos/X, gráficos/gtk, etc. masqdialer permite establecer permisos de acceso por computadora y por usuario por lo que puede configurarse bien quien gasta teléfono en qué servidor y quien no.

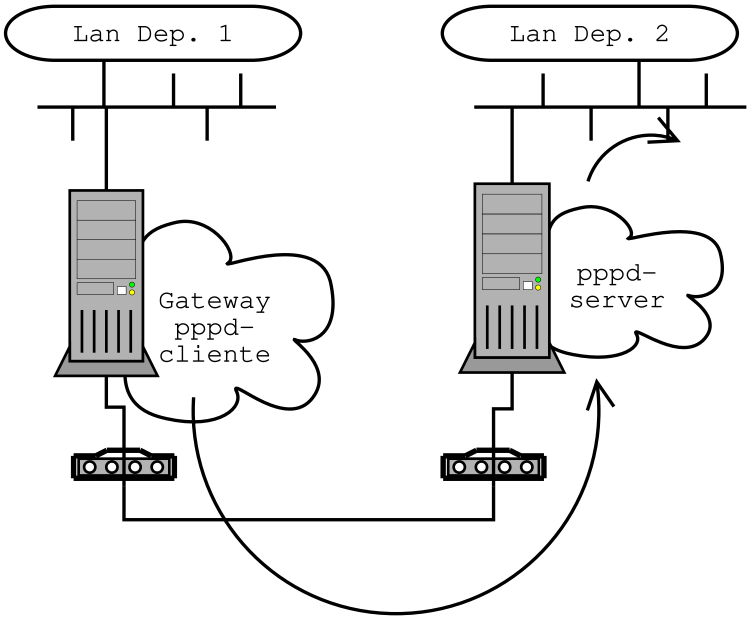
Apletts Hay varios utilidades para X-Windows, que consisten en esencial en un botón y una "luz". Haciendo Click en el botón establece/corta el enlace. La luz cambia de color si se está conectado.

Desde adentro

algunos conceptos que hacen pppd interesante para redes locales.

ip-up/ip-down Muchas veces se requiere realizar algunas tareas específicas al momento de establecer un enlace con la red: en primer lugar para establecer rutas especiales, reglas firewall para el interfáz u otras configuraciones especiales de la red, pero también por ejemplo para enviar automáticamente todo el correo en cola, y para bajar correo o noticias del internet, cuando el enlace se establece. Para esto pppd puede ejecutar un script al establecer el enlace, y otro al cerrarlo. En las instalaciones modernas este script accede a un directorio /etc/ppp/ip-up.d y /etc/ppp/ip-down.d en el cual pueden ubicarse de manera ordenada diferentes scripts, que se ejecutan en orden específico (alfabético). El mecanismo es parecido al que se utiliza con "init" del sistema V de unix. De esta manera cada paquete de software puede agregar y quitar tareas según necesidad y sin interferir con otros paquetes.

/etc/ppp/peers /etc/chatscripts Para establecer un enlace tienen que compaginar el discado y la configuración del pppd. En el sistema Debian se encuentran los scripts chat para la primer tarea bajo el directorio /etc/chatscripts, y la configuración del pppd bajo /etc/ppp/peers, bajo el mismo nombre. pppconfig modifica ambos archivos al configurar un enlace nombrado. Eventuales cambios manuales pueden realizarse allí mismo.



Ejemplo de uso de ppp para una red local

pppoe es el acrónimo para PPP Over Ethernet. En este caso se utiliza una conexión Ethernet para el enlace transparente.

pptp es el acrónimo para Point to Point Tunnelling Protocol. Es usado para establecer una conexión bajo DSL/ADSL ([Asyncronios] Digital Subscriber Line).

ippp es el protocolo ppp utilizado por conexiónes ISDN (Integrated Services Digital Network).

y otros ppp es muy versátil para establecer conexiones TCP/IP de punto a punto. Por lo tanto es utilizado muchas veces para redes WAN. El “costo” de desarrollar la tecnología entonces se reduce a la tarea de establecer un canal transparente. Nosotros podemos hacer lo mismo. Por ejemplo pueden enlacarse dos departamentos distantes de una institución mediante la planta telefónica, o eventualmente con un simple cable de dos hilos que simula una línea telefónica (requiere alimentación electrica). En el último caso nos podemos valer del hecho, que los modem intentan establecer una conexión con el comando “ATD”, aunque no especificamos un número telefónico. Por otro lado, con el comando ”ATA” podemos iniciar la negociación de velocidad, aunque no se reciba la señal de marcado (“RING”). Para que el modem primero no corte la línea por falta de tono (“NO DIALTONE”) se puede omitir esta condición con el comando “ATX3”. En la imagen 7.5.1 se muestra solamente una rama de la configuración. La computadora a la izquierda utiliza (al arrancar) el modem y ppp para tratar de conectarse a la computadora a la derecha. Todas las computadoras en la Lan del departamento 1 utilizan esta computadora como Gateway para la Lan del departamento 2. Si el enlace no se puede establecer, la computadora a la izquierda pone el modem en modo answer ("ATA") con un timeout infinito. La misma configuración se realiza con la segunda computadora.

demand dialing, dialdcost El programa diald (dial-daemon), utiliza un enlaces virtual (en este caso SLIP), para detectar si alguna computadora de una LAN solicita una conexión con Internet. En este caso establece un enlace ppp con el modem y enruta la conexión a través de ella. Si durante un cierto tiempo no hay tráfico la línea telefónica se corta. De esta manera se puede tener un enlace casi contínuo, sin tener que pagar el costo de una línea dedicada, los usuarios solamente tienen que tener un poquito de paciencia para permitir que se establece el enlace.

Junto con configuraciones de este estilo vienen práctico utilidades que permiten sumar el costo total de la conexión, llevar el control de cuando se establece y cuando se corta la conexión, y hasta que usuari@ ha consumido cuánto tiempo de conexión. El programa dialdcost y ip-account se occupan de estas tareas.

7.5.2. Syncronización de una Portatil con una PC de Escritorio

En vez de utilizar disquetes, o unidades externos de almacenamiento de datos como ZIP-Drives etc. para copiar datos entre una computadora portatil y su PC "de casa" y de oficina se pueden enlazar mediante una conexión de cable. Hay soluciones en MS-Dos y Windows, que siguen un esquema de maestro – esclavo donde una parte inicia la comunicación y domina el proceso de copiado. El programa posiblemente más antíguo es "kermit", que utiliza un puerto serial en ambas computadoras. El puerto serial tiene la desventaja de una velocidad baja y la serialización de los datos: un caracter requiere aproximadamente 10 ciclos para su transmisión.

Una salida de este cuello de botella es el uso del puerto parallelo, desvirtuando las funciones de las líneas de control. La idea es, utilizar cuatro líneas de datos (de los ocho disponibles) como salida parallela, y cuatro de las líneas de señalización como entradas. Estas líneas normalmente usa la impresora conectada para señalizarle a la computadora fallas como la falta de papel o su estado para recibir datos (occupado/listo). Las "entradas" se conectan a traves de un cable especialmente preparado a las "salidas" de otra computadora. Ambas computadora por supuesto utilizan un programa especial, con el cual pueden intercambiar datos serializados en nibbles. Haciendose común este arreglo fue tomado en cuenta por los diseñadores del hardware de las computadoras y actualmente hay puertos parallelos que pueden reconfigurar las ocho líneas de datos como entrada y/o salida y se pueden utilizar las líneas de señalización para el control. El hardware permite de esta manera a veces inclusive transferencia de bloques grandes de datos mediante DMA, aliviando la CPU del trabajo de controlar la conexión. Tambien esta forma de conexión utiliza un cable especial, de ningúna manera puede usarse un cable de impresora para interconexión de dos puertos parallelos.

Linux provee una alternativa al enfoque maestro – esclavo, proveyendo un dispositivo de red a través del puerto parallelo. El protocolo utilizado permite el transporte bidirecional de paquetes TCP/IP a través del puerto parallelo y se llama PLIP: Parallel Line Interface Protocol. El driver (módulo de núcleo) y el nombre del dispositivo de red llevan este mismo nombre.

Es recomendable que el driver se compilado como módulo de núcleo, y no integrado fijamente a este, ya que de esta manera se puede cargar y descargar de la memoria y liberar el puerto parallelo después de usar la conexión PLIP. Si se compila fijamente, no será posible utilizar el puerto parallelo para otra cosa, como por ejemplo la impresión[2].

7.5.3. Preparación del puerto paralelo para PLIP

PLIP requiere el uso de la interrupción hardware para el puerto paralelo. Normalmente este no está activado, ya que el uso del puerto paralelo para la impresión no lo requiere, y las interrupciones son un recurso muy valioso como para ocuparlo innecesariamente. Por lo tanto necesita activarse la interrupción, o a la hora de cargar el módulo del driver del puerto paralelo (parport_pc), o posteriormente mediante el sistema virtual de archivos 'proc'.

> Notese, que hay (por lo menos) dos cambios esenciales entre las versiones de núcleo antes de y apartir de la versión 2.4. respectivamente. El primero es la ubicación de los módulos de núcleo en una jerarquía de directorios en /lib/modulos, y la segunda es la estructura de la información del núcleo en el sistema virtual de archivos /proc. A continuación se presenta la información pre-2.4. Para núcleos 2.4 y adelante hay que cambiar los parámetros.

driver: insmod /lib/modules/<version-del-núcleol>/misc/parport_pc io=<puerto> irq=<número>

proc: echo <número> > /proc/parport/<número-puerto paralelo>/irq

donde:

<número> es el número de la interrupción que utiliza el hardware y

<puerto> la dirección de I/O (entrada/salida) correspondiente.

En la siguente tabla se dan algunos valores estándard:

Puerto	I/O	Irq	nombre en dos
/dev/lp0	0x378	7	LPT1:
/dev/lp1	0x278	5	LPT2:
/dev/lp2	0x3bc	-	En tarjetas monocromaticas

A esta altura tenemos un dispositivo de red disponible (se puede visualizar en /proc/net/dev), normalmente llamado "plip0". El próximo paso es:

[2] En los núcleos Línux más modernos ya existen mecanismos para el uso múltiple del puerto paralelo.

7.5.4. Configuración del enlace TCP/IP

Por supuesto tiene que efectuarse el anterior paso con éxito en las dos computadoras que se pretenden comunicar. A continuación hay que configurar en ambas el interfáz plip0 para que pueda comunicarse con la otra computadora. En la mayoría de los casos el enlace será temporal por lo que utilizamos números IP privados. Si queremos conectar una portatíl en diferentes computadoras de mesa, p.ej: en casa, en el trabajo, en la universidad, donde un cliente, etc. una posible configuración es, configurar en todas las computadoras de mesa el mismo número IP, digamos que sea: 192.168.255.254. La portatíl se configuraría con un número fijo, sea: 192.168.255.253, y se puede facilitar el trabajo dandole un nombre a estos números IP en el archivo /etc/hosts, p.ej plippc para las computadoras de mesa y pliplap para la portatil respectivamente.

Los enlaces plip tienen que configurarse con la opción punto a punto:

```
ifconfig plip0 192.168.255.253 pointopoint 192.168.255.254
```

en la portatil y

```
ifconfig plip0 192.168.255.254 pointopoint 192.168.255.253
```

en la(s) computadora(s) de mesa respectivamente.

Para más completo aquí una sección del archivo /etc/hosts, que sería iqual para todas las computadoras

```
192.168.255.253 pliplap
192.168.255.254 plippc
```

Una configuración ejemplar que funcionaría con la mayoría de las computadoras actuales puede ser:

```
echo 7 >/proc/parport/0/irq
modprobe plip
ifconfig plip0 192.168.255.253 pointopoint 192.168.255.254
```

Este segmento de comandos se puede grabar en una archivo script, p.ej. llamado plipon y ejecutar cuando se quiere conectar las dos computadoras. Obviamente hay que modificar la última línea de acuerdo a qué máquina se trata.

7.5.5. Observaciones adicionales

En caso que no se puede iniciar el uso del puerto como interfáz plip, puede ser que hay trabajos de impresión pendiente y el daemon de la cola de impresión "lpd" está occupando el puerto. Este hecho se puede revisar por ejemplo con el comando lpq, o lpstat. Si hay trabajos en cualquier cola de impresión que utilize el puerto de impresión en cuestión (revisar /etc/printcap y las entradas :lp=<puerto>:) tienen que terminarse o cancelarse estos primero.

El puerto paralelo puede existir con varias capacidades:

SPP es el modo estándard compatible con *todas* las computadoras PC (Standard Parallel Port)

EPP es un modo mejorado (Enhanced Parallel Port), en el cual la computadora puede leer informaciones de identificación de una impresora conectada (y compatible con IEEE 1284)

ECP prove mas "capacidades" (Extended Capabilities Port)

Si las dos computadoras disponen de puertos paralelos con dispositivos mejorados se puede posiblemente utiliza el modo de transferencia de un byte completo llamado también Modo 1. Para esto se requiere de un cable especial, que solamente se debe utilizar con las dos computadoras configurados para este modo, ya que de otra manera podrían causarse daños en los dispositivos de hardware. En resumen: El modo estándard, o Modo 0 utiliza un cable "especial", llamado cable "Parallel-Null-Modem" (que es el mismo que utiliza LapLink para MS-DOS/Windows), y que no puede dañar al hardware (del puerto de impresión), pero por supuesto no puede usarse para imprimir. Este modo es lento ya que solo transmite un nibble - medio byte a la vez. El modo más rápido, Modo 1, requiere de: puertos mejores que SPP en ambos lados, un cable "especial" y que las computadoras estén configuradas ambas en Modo 1. La información como hacer estos cables se encuentra p.ej. en los archivos fuente del núcleo en el subdirectorio Documentation/networking/PLIP.txt

7.5.6. Intranet con LAN/Ethernet

8. Impresión

8.1. LprNg e impresión remota

8.2. Cupsys

8.3. Ghostscript

8.4. Magicfilter

8.5. Redhat-Printtool

[opcional]

9. Redes Heterogeneas

9.1. Redes Microsoft - Samba-Suite

9.1.1. Conectar Linux a un servidor SMB

9.1.2. Proveer Servicios SMB desde Linux

9.2. Redes Novell - ncp

[opcional]

9.3. Redes Appletalk netatalk

[opcional]

10. Usuarios

10.1. Cuentas especiales, root, sysadmin

10.2. Cuentas normales

10.3. Grupos de trabajo

10.4. Authorización en la red

[opcional]

10.5. Como no ser root

[Recetas para adminstrar un sistema Linux en la vida diaria. Instalación de herramientas sudo, calif, etc., importante pero opcional.]

11. Linuxconf

[herramienta homogénea de configuración]

Parte III.

Instalación

12. Proceso de arranque

[Qué elementos tiene un sistema operativo corriendo,

Cómo se llega a este estado -> mapa para las herramientas de instalación del sistema

]

13. Disquetes de arranque y rescate

[Comparación Debian, Redhat, syslinux, lilo y autoarranque del núcleo]

14. Disco duro

[Preparación del disco duro, planificación de particiones]

14.1. Partitiones y sistemas de archivos

[fdisk, y formateo, swap-partitions]

14.2. Lilo y grub

[grub solo opcionalmente. Lilo, instalación, configuración]

15. Instalación del Software

[Instalación del sistema base. Qué son paquetes precompilados de Software]

15.1. Redhat

[opcional]

15.2. Debian

[Instalación CD y Red]

15.3. Perfiles, Paquetes, Fuentes

[Perfiles de computadoras según su uso. Selección predeterminada de paquetes. Instalación via mirrorring.

Dependencia de paquetes, problemas con incompatibilidad y versiones, Instalación desde el código fuente.]

16. Hardware específico

En este capítulo se explica, como se puede activar dispositivos que no están incluidos en la configuración básica de las distribuciones. También se indica la forma como optimizar y reinstalar el núcleo del sistema operativo mismo.

16.1. Núcleo modular

Una fundamental característica del núcleo de Linux es su modularidad. Mediante el mecanismo de módulos del núcleo se pueden conseguir dos efectos:

- Se reduce el tamaño de la imagen del núcleo (el archivo que se carga al arrancar el sistema operativo)

- Se pueden re-inicializar driver para ciertos dispositivos de hardware, o usar consecutivamente diferentes drivers para el mismo dispositivo sin recargar todo el núcleo (sin reiniciar la computadora).

16.1.1. Objetos dinámicos

En la programación se utilizan bibliotecas de programas para resolver tareas comunes. P.ej. una de las más famosas biblioteca en la programación en C es asociada con el archivo de inclusión <stdio.h>. Es la biblioteca que contiene las rutinas para imprimir y leer caracteres desde archivos y dispositvos.

A la hora de escribir un programa, el o la programador/a ya no tienen que reinventar las rutinas de entrada y salida, sino utilizan las prefabricadas de la biblioteca.

Como casi todos los programas en el sistema incluyen esta biblioteca de rutinas, a la hora de cargarlos se desperdicia espacio de memoria, porque el código de programa de las rutinas de la biblioteca está incluido en cada programa. Para evitar esta desventaja se inventaron las bibliotecas dinámicas, u objetos compartidos. El primer nombre (DLL: dinamic linked library = biblioteca enlacada dinámicamente) proviene de los sistemas operativos Microsoft, mientras en Linux se utiliza la segunda expresión: Shared Objects = objetos compartidos, designados con la extensión de archivo ".so". Las bibliotecas absolutas, o estáticas, que se requieren solamente para la compilación tienen la extensión ".a".

En la compilación de programas estáticos se incluyen los archivo de las bibliotecas absolutas después de generar el código del programa mismo, y a continuación se resuelven los saltos a las rutinas de la biblioteca mediante un programa llamado el Linker =

enlazador. El resultado es un archivo monolítico que contiene todo el código necesario para ejecutar su tarea.

En contraste se compilan programas modulares sin incluir el código de las bibliotecas. El Linker ahora tiene un rol diferente, y en efecto es un programa diferente, en Linux: "ld.so", shared object linker. A la hora de cargar un programa con bibliotecas dinámicas ld.so entra en acción, analiza cuáles bibliotecas requiere el programa para funcionar, los carga en espacios libres en la memoria y en este momento modifica el código del programa de tal manera, que todos los enlazes resuelven correctamente. Si la biblioteca ya está cargado en la memoria, ld.so no vuelve a hacerlo, sino resuelve los saltos hacia las rutinas correspondientes hacia la biblioteca presente. Esto requiere, sin embargo, que las bibliotecas estén escritas de una forma que permite reutilizar las mismas rutinas por diferentes programas.

Ahora comparemos módulos estáticos con módulos dinámicos:

Ventajas de módulos dinámicos:

- requieren menos memoria, total, si mas programas las comparten
- si una biblioteca tiene una falla, se puede corregir sin necesidad de recompilar todos los programas. Solo se sustituye la biblioteca respectiva.

Desventajas de módulos dinámicos:

- Las veces que se carga un programa, se tiene que resolver de nuevo las direcciones: pérdida de tiempo.
- Si se copia un programa a otra computadora, también tienen que copiarse todos los archivos de bibliotecas de los cuales depende.
- cada programa y cada biblioteca crece un poco de tamaño, porque tiene que contener información adicional para su enlace dinámico exitóso.
- Se requiere el programa ld.so (y su adecuada configuración) para poder ejecutar un programa.

En el sistema Linux es necesario ejecutar la utilidad "ldconfig" como root, cuando se instalan nuevas bibliotecas. El archivo "/etc/ld.so.conf" contiene una lista de directorios que se utilizan para encontrar y registra bibliotecas dinámicas. Esta lista solamente es para directorios adicionales, ya que ld.so "sabe" las vías de acceso estándard.

16.1.2. Módulos del núcleo

El núcleo del sistema operativo es por un lado también un programa C, pero por otro lado es especial, en el sentido que se carga desde "la nada", y no puede contar con muchas de las utilidades versátiles, que el sistema operativo pone a disposición. Sin embargo se integró al núcleo de Linux un sistema de resolución dinámica de "bibliotecas", que en este caso se llaman módulos. Esto quiere decir, que ciertas rutinas del núcleo no se incluyen en el mismo archivo que el bootstraploader - el cargardor inicial lee del disco duro (o de donde sea), sino que residan en el disco duro en archivos aparte. A la hora que se requiere estas rutinas se tienen que copiar estos módulos del disco duro a la memoria principal, y resolver las direcciones dentro del núcleo a la ubicación que tengan en este momento. En la figura 16.1.2 se ilustra la diferencia entre un núcleo estático y un núcleo con módulos.

Usar un núcleo con módulos implica, que primero tiene que funcionar el acceso al disco duro (o más preciso: al sistema de archivos) para que se pueden cargar módulos adicionales y por lo tanto que las rutinas esenciales para acceder al sistema de archivo no pueden ser compilados como módulos. Además el núcleo necesita "saber" cuáles módulos pueden potencialmente cargarse "en el". A la hora de compilar el código fuente del núcleo de Linux se le indica, cuales rutinas son residentes (fijos) en el, cuáles se compilan como archivos de módulos y cuales no se compilan. Para los que son módulos se reservan los saltos para cargarlos en el núcleo, por lo que se agrandece mínimamente el volumen del núcleo.

Ventajas de módulos de núcleo:

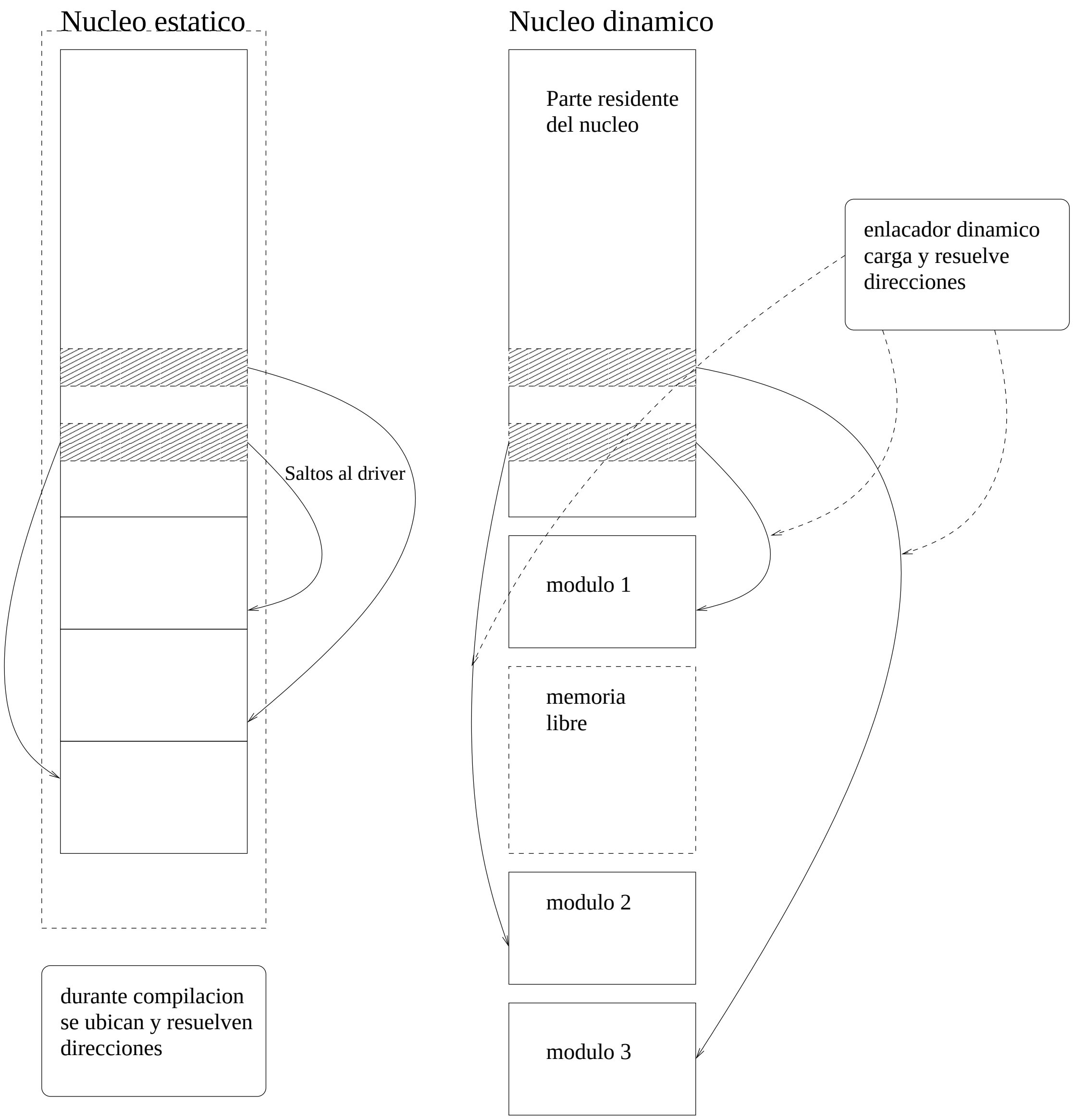
- Se puede reducir el volúmen del archivo de núcleo:
  - así es posible grabar un archivo núcleo en un disquete, que tiene menos capacidad de almacenamiento que el núcleo total.
  - el núcleo reducido cuenta con un tiempo de carga reducido, aunque este tiempo se invierte (y con interéses) a la hora de cargar el núcleo.
- Se puede mejorar el rendimiento de la memoria virtual, descargando rútinas del núcleo de la memoria, que no se requieren en un determinado momento.
- Se puede descargar y volver a cargar módulos, y de esta manera re-inicializar dispositivos
- Se puede usar un dispositivo de diferentes maneras. Ej.: el puerto paralelo puede usarse para plip, impresión, discos externos (zip-drives)[1]. Para esto se descarga el módulo en uso, y después se carga el otro.

Los mismos módulos pueden ser "modulares" por ellos mismos, quiere decir, si varios módulos utilizan las mismas rutinas, estos pueden separarse en un módulo aparte, y se dice que entonces estos módulos dependen del último. Tiene que cargarse primero el módulo con las rutinas compartidas, para poder cargar los otros.

El núcleo puede ser configurado también para cargar automáticamente los módulos que necesita. Esta funcionalidad se llama kernel-daemon o kerneld.

Hay varias utilidades para cargar, descargar y administrar los módulos :

[1] En el caso del puerto paralelo está resuelto el uso múltiple de una manera mas elegante en los núcleos de versión 2.2 y mayores. Hay un módulo para el puerto, el cual es usados por otros módulos. Si un módulo accede al módulo del puerto bloquea los otros, cuando termina de usar el módulo lo cede a los demás módulos. No se necesita descargar y recargar los módulos individuales.



Diferencia entre núcleo estático y núcleo modular.

insmod <via de acceso> carga un archivo módulo de núcleo en la memoria. La extensión de estos archivos es ".o". En la líne de comando se especifican los parámetros que son específicos para este módulo.

lsmod alista todos los módulos actualmente cargados en la memoria.

rmmod <nombre de módulo> remueve un módulo del núcleo. El nombre del módulo es comunmente el nombre del archivo, pero sin la extensión ".o".

modinfo <via de acceso> muestra información sobre el módulo indicado por <via de acceso>. En especial se puede averiguar con este comando, cuales parámetros son acceptados o requeridos por el módulo. También se puede utilizar el nombre del módulo en vez de la vía de acceso.

Y existe todo un sistema de gestión de módulos, llamado modutils, que facilita poner en orden las dependencias de los módulos. En un sistema instalado, estas utilidades por lo general no se necesitan invocar manualmente, porque están integrados en el arrance y la administración automático.

depmod crea un archivo de dependencias para todos los módulos en el disco duro.

modprobe <nombre de módulo> modprobe utiliza esta lista para cargar el módulo indicado por <nombre de módulo>, pero carga anteriormente todos los módulos del cual depende el indicado.

En el archivo /etc/modules.conf pueden especificarse además varios parámetros que se aplican a la hora de cargar un módulo. Si se trata de módulos que acceden directamente a un dispositivo de hardware, muchas veces se requiere especificar el puerto de Entrada/Salida o la interrupción que utiliza el dispositivo. Especialmente si pueden haber varios dispositivos del mismo tipo en una computadora, se puede especificar el orden en el cual se les asigna su numeración. Otro comando utilizado con frecuencia en este archivo es "alias" que permite darle un "nombre" sinónimo a un módulo. Así se puede por ejemplo llamar al módulo del hardware de sonido siempre "sounddriver", independientemente si es un soundblaster, o si es un chipset cmpci.

El comando "insmod" requiere, que se especifíca las opciones en la línea de comandos, mientras modprobe los lee desde el archivo modules.conf. Con el tiempo la cantidad de módulos ha crecido a un gran número, así que el sistema Debian Gnu/Linux utiliza un mecanismo de diversificación de estos archivos. En el directorio /etc/modutils se pueden crear archivos o editar los existentes, en los cuales se especifican opiones o comandos correspondientes a modules.conf. Con el comando "update-modules" todos estos archivos son integrados en el archivo modules.conf automáticamente, el cuál no debe ser editado a mano.

En el sistema Debian Gnu/Linux se utiliza el archivo /etc/modules para indicar, cuales módulos deben cargarse automáticamente al iniciar el sistema. Se escribe una línea por cada módulo, solamente con el nombre del módulo que se quiere cargar.

Ya que un sistema puede trabajar con diferentes versiones de núcleos existe un esquema que permite a los modutils identificar cual módulo cargar. Los módulos están ubicados en el directorio /lib/modules, en un subdirectorio con el nombre de la versión del núcleo, para el cual fueron compilados. Si tenemos un núcleo de la versión 2.2.17 (Debian 2.2) y uno de la versión 2.4.0 entonces encontramos dos directorios: /lib/modules/2.2.17 y /lib/modules/2.4.0[2].

Bajo estos directorios hay una jerarquía de subdirectorios, para dividir los archivos de módulos según su función o tarea. Para detectar cual versión de núcleo estamos usando puede ejecutarse la línea de comando: uname -a, o uname -r

16.2. Recompilación del núcleo

Pueden haber varias razones por qué recompilar el núcleo del sistema operativo, que parten todos del hecho que el núcleo "universal" o genérico que se instala en el proceso de la instalación del sistema no es adecuado para la tarea que se quiere realizar con la computadora. Pueden haber uno varios de los siguientes escenarios:

- La computadora utiliza dispositivos de hardware para los cuales no existían los gestores en el tiempo cuando fue fabricado el núcleo genérico.
- La computadora utiliza dispositivos de hardware pocas veces usados y que no son habilitados en el núcleo genérico.
- Se quiere compilar estáticamente un gestor que está actualmente compilado como módulo o vice versa.
- La aplicación de la computadora requiere facilidades o parámetros especiales, que no son habilidados en el núcleo genérico.
- Se quiere instalar un núcleo más reciente para utilizar funcionalidades no incorporadas en el núcleo genérico.
- Se quiere reducir el tamaño del núcleo y/o la cantidad de módulos, solamente incorporando los gestores necesariamente requeridos para el sistema/la aplicación. Esto puede ser útil para liberar memoria para las aplicaciones, o puede ser necesario si la computadora dispone de muy poca memoria principal.

El procedimiento a realizar ha llegado a ser bastante estándardizado y no requiere conocimientos de programación. El primer paso es la instalación del código fuente del núcleo, por lo general (y recomendablemente) en un subdirectorio de "/usr/src". En el sistema Debian el núcleo se instala en un subdirectorio llamado "kernel-source-<version>", en RedHat se utiliza "linux-<versión>" y se crea un enlace simbólico "linux" al subdirectorio del núcleo actualmentente utilizado.

[2] Si compilamos dos versiones diferentes de un núcleo de la misma serie, tenemos que distiguirlos y también hay que usar directorios diferentes para los núcleos. Eso es posible con un anexo al número de versión, que se especifica cuando se compila el núcleo.

Para trabajar con el núcleo hay que entrar en este directorio, que se llama también el directori raíz de la fuente del núcleo. El siguiente paso es la configuración de todos los parametros y la definición de los gestores estáticos y/o módulos de núcleo que se quiere incluir. Para esto hay tres utilidades:

1. Modo texto solamente: "make config"
2. Modo diálogo, con un interfáz gráfico en la consola de texto: "make menuconfig"
3. Modo gráfico, con un interfáz Tcl/Tk en X-windows: "make xconfig"

La primera versión no es recomendable, ya que procesa todas las preguntas de configuración de forma secuencial y no permite regresar a decisiones anteriores. La segunda y la tercera versión permiten seleccionar las categorias interactivamente y repetir los pasos de configuración cuantas veces sea necesario. Además permiten una cierta administración de configuraciones, ya que permiten cargar y salvar archivos de configuración nombrados. Es perfectamente posible y a veces inclusive recomendable compilar en una computadora núcleos para otras computadoras. El código fuente actualmente alcanza los 80 Megabyte de espacio y se requier alrededor de un cuarto a una media hora con una computadora veloz con bastante memoria para compilarlo. Así que es preferible compilar núcleos para computadoras lentas en otras computadoras y después copiarlos a la máquina destino. Para esto se maneja p. ej. archivos de configuración con los nombres de las computadoras en la computadora donde se compilan los núcleos, para más rápido acceso a las configuraciones diferentes.

Al finalizar la configuración se graba un archivo ".config" que contiene todos los parametros de compilación necesarios. Ahora se procede a compilar el núcleo, esencialmente con los comandos:

```
make deb; make clean; make vmlinuz; make modules; make modules_install
```

Se utiliza preferiblemente una sola línea de comando, ya que cada uno de los procesos es tardado y de esta manera no se tiene que estar presente para dar los comandos individuales a concluir cada paso. Sin embargo, si la ejecución de un paso se interrumpe por un error, un hecho que pocas veces ocurre, se tiene que reinicializar solamente a partir del comando fallado.

La compilación del núcleo es una prueba dura para el procesador y para la memoria principal de la computadora. Es casí imposible que una memoria con fallas no se descubra a la hora de (re)compilar el núcleo. El símptoma presentado en este caso es muchas veces un mensaje raro con "broken pipe" o "compiler error", o puede ocurrir en la fase de generación del código de máquina, que normalmente nunca falla. Con otras palabras: Si duda de la memoria de su computadora compila un núcleo de Linux en ella. Otra razón para la compilación que no mencioné arriba.

Para usuarios de un sistema Debian Gnu/Linux las cosas se facilitan con el paquete utilitario "kernel-package", que provee el comando "make-kpkg". Para compilar un núcleo y preconfigurarlo para la instalación solamente se necesita ejecutar:

```
make-kpkg kernel-image
```

y eventualmente de antemano “make-kpkg clean”.

16.2.1. Parametros de configuración

Hay varios categoría de configuraciones para un núcleo, las cuales cada una tiene su cantidad de opciones. Muchas de las categorías o son presentes, o no lo son. En caso que se selecciona por ejemplo compilar un núcleo sin soporte de red, ya no se hacen preguntas acerca de tarjetas de red, protocolos de red, o sistemas de archivos que permiten compartir archivos en red.

A continuación se describe brevemente cada categoría de configuración del núcleo versión 2.2.17 que es estándard para la distribución Debian Gnu/Linux. Esto está tomado de la configuración mediante “make menuconfig”. En el programa de configuración mismo se puede obtener ayuda extensiva para cada categoría y para cada pregunta (opción), pero también hay mucha infomración en el código fuente del núcleo mismo. En primer lugar esta se encuentra en el directorio “Documentation” dentro de la raíz del núcleo, después hay información adicional en los directorios de los archivos .c mismos.

Code maturity level options permite activar o desactivar opciones en todas las otras secciones, que no se consideran todavía estábles. Para considerar un driver, o una opción de configuración como estable tiene que pasar un tiempo de prueba considerable, así que muchas veces vale la pena habilitar esta opción, ya que provee gestores considerablemente estables.

Procesor type and features en esta sección se selecciona el tipo de procesador para el cual se compila el núcleo. Si se quiere compilar un núcleo genérico posiblemente se considera seleccionar un procesador 486 o máximamente Pentium. Seleccionando un procesador más avanzado optimiza el código creado, pero el núcleo resultante no puede correr en un procesador inferior.

Loadable module support aquí se habilita el uso de módulos de núcleo. En principio se pueden utilizar los mismos archivos módulos para diferentes versiones de núcleo, aunque eso no siempre funciona. En esta sección se define si solamente se permite cargar módulos con la versión precisa del núcleo, que requiere un chequeo adicional (y una pequeña pérdida de espacio y tiempo) o si se carga los módulos sin chequeo.

General setup en esta sección se activa y desactiva diferentes subsistemas, como redes, soporte para diferentes buses de periféricos (PCI, MCA), etc.

Plug and Play support habilita soporte para la autodetección y/o autoconfiguración de puerto de Entrada/Salida de los dispositivos de hardware en buses ISA y PCI. Nota: Plug and Play o PnP (Enchufa y juega) del bus ISA a veces también se traduce con Plug and Praise (Enchufa y comienza a rezar).

Block devices permite seleccionar los gestores para floppy, disco duro y otros dispositivos de almacenamiento de datos. También se puede habilitar aquí el uso de múltiples discos en configuraciones RAID, que provee aumento de capacidad, aceleración del acceso, o seguridad mejorada en los discos duros.

Networking options configura el soporte para diferentes facilitades y protocolos de red, y opciones para adaptar los gestores a computadoras muy rápidas o muy lentas. En especial, aquí se activa el soporte para IP-Masquerading (NAT) y para Firewall, opciones que permiten configurar la computadora como servidore especializados en conectar una red local al Internet.

Telphony Support para activar el gestor para dispositivos de telefonía.

SCSI support para activar el soporte para diferentes tipos de dispositivos SCSI (discos, cintas, CDROM, otros), y para seleccionar los gestores para la tarjeta SCSI que se utliza. Si se quiere usar ZIP-Drives de Iomega que utilizan el puerto paralelo se tiene que utilizar esta sección, ya que estas unidades utilizan una simulación del puerto SCSI via el acceso al puerto paralelo.

I2O device support I2O es una tecnología de dispositivos genericos de entrada/salida que relevan la CPU del trabajo de acceder directo a los controladores físicos y aumentan la velocidad total del sistema. Esta opción solamente se requiere si la computadora utiliza este tipo especial de hardware.

Network device support para seleccionar los gestores para los dispositivos de red en la computadora.

Amateur Radio support esta sección ofrece la selección de los diferentes protocolos de red que se utilizan para el intercambio de datos via packet radio, y los gestores para los dispositivos disponibles de esta forma de conexión.

IrDA (infrafred) support gestores para los enlaces de infrarojo.

ISDN subsystem Integrated Services Data Network es una norma ampliamente usado en Europa que facilita varios canales de intercambio de datos y dos canales de audio en paralelo mediante la línea telefónica.

Old CD-ROM drivers (not SCSI, not IDE) hay algunas unidades lectoras de CD-ROM que utilzan tarjetas especiales para su control, en esta sección se encuentran los gestores para ellos. Una sección que prácticamente nunca se utiliza.

Character devices Los gestores para todos los dispositivos de carácter. En especial para la console (monitor y teclado), los puertos seriales y los puertos de teletype virtuales "pty" que no corresponden a un dispositivo de hardware, estos se usan extensamente para el sistema X-Windows. También se encuentran aquí los dispositivos para captura de televisión.

Filesystems en esta sección se seleccionan, cuales sistemas de archivos serán disponibles para el núcleo. Se pueden seleccionar diferentes sistemas de archivos para medios físicos, y sistemas de archivos en red, o sea, gestores que permiten acceder a archivos remotos de una manera transparente, como si estos estuvieron montados en un directorio local.

Console drivers prove opciones para las tarjetas de video y para activar el soporte para Frame buffers devices. Estos últimos son dispositivos virtuales, que proveen una presentación equivalente de la pantalla independientemente de la arquitectura (el procesador) del sistema.

Sound permite seleccionar la tarjeta de sonido que se está usando.

Kernel hacking activa una opción para programadores de núcleo, que permite ver los registros del procesador mediante una combinación especial de teclas.

Load an Alternate Configuration File carga un archivo de configuración nombrado

Save Configuration to an Alternate File salva la configuración actual en un archivo nombrado.

En la versión 2.4. del núcleo la estructura es un poco diferente, aquí solamente se discutan opciones no existentes en el núcleo 2.2.

Memory Technology Devices (MTD) soporte para memoria Flash, etc. esto es memoria no volátil de grán volúmen que puede usarse como “discos estáticos”.

Parallel port support para el driver genérico, en dependencia de la arquitectura de computadora, el cual va usado para el gestor de impresión, discos paralelos, Plip, etc.

Multi-device support (RAID and LVM) vea la opción “Block devices” arriba.

ATA/IDE/MFM/RLL support El soporte para diferentes gestores de nivel bajo de discos fue separado en esta sección, en 2.2. esta en “Block devices”.

Fusion MPT device support para computadoras que usan este dispositivo. Es una tecnología que conecta los dispositivos de entrada/salida mediante fibra óptica a la CPU.

Input core support habilita el soporte para dispositivos USB de interacción human - teclados, joystick, etc.

Multimedia devices para dispositivo de procesamiento de señales de video; en 2.2.17 estas opciones están en “Character Devices”.

USB support gestores para los dispositivos hardware de USB = Universial Serial Bus, una tecnología que permite concatenar teclados, ratones, impresoras, escaner, discos externos etc. en un solo cable serial.

Bluetooth support son gestores para esta clase de dispositivos USB.

16.2.2. Versiones de núcleos

16.3. Instalación de un núcleo

En un núcleo modular hay varios componentes que poner en su lugar para que el sistema pueda utilizarlo en el arranque. Primero lógicamente el archivo de núcleo mismo, y un bootstrap loader que lo carge a la memoria en el inicio del sistema. Este va a ser lilo en nuestro caso. Después se necesitan copiar los archivos de los módulos correspondientes al núcleo al directorio respectivo. y por fín conviene copiar el archivo con los símbolos de rutinas exportádos del núcleo a su lugar correspondiente, porque varias utilidades administrativas los utilizan para mostrar información sobre el sistema.

En la misma computadora se puede utilizar varios núcleos, inclusive utilizando el mismo dispositivo para el sistema de archivos raíz.

16.3.1. Lilo

Aunque en teoría se puede utilizar cualquier lugar en el sistema de archivos que esté disponible al Bios de la computadora para guardar el archivo del núcleo, existen convenciones que preferiblemente se respetan. Hay una convención de tener un enlace simbólico con nombre “vmlinuz” en el directorio raíz. El directorio “/boot” se utiliza para guardar todos los núcleos que se ofrecen para el arranque. El núcleo estándard se instala entonces con un archivo /etc/lilo.conf estándard que contenga las líneas:

```
image=/vmlinuz
     label=Linux
     read-only
```

y mediante el comando: “lilo”

Una convención práctica del sistema Debian Gnu/Linux es, tener un segundo enlace simbólico /vmlinuz.old que indica al núcleo genérico de instalación en el directorio /boot y que se puede cargar con la etiqueta “LinuxOld”. De esta manera se garantiza que siempre hay un núcleo “de reserva” si se compila e instala un núcleo nuevo que no funciona.

16.3.2. Módulos

Con el comando “make modules_install” ya se copian los archivos de módulos generados a la hora de compilar el nucleo en su lugar correspondiente. No es trivial ubicar dónde específicamente va a parar un archivo de módulo. Entre las diferentes versiones de núcleos se han estado desarrollando también diferentes formas de organizar y repartir los archivos de núcleos.

Las versiones 2.0 del núcleo han sido orientados a computadoras con procesadores intel (i386) y contemplando solamente hardware “estándard”, más que todo controladores de discos (dispositivos bloque), de terminales (dispositivos de caracteres), dispositivos de red y algunos mas. La gran variedad de hardware, por ejemplo dispositivos de sonido ha requerido separar estos en un subdirectorio aparte en el núcleo 2.2.

A partir del núcleo 2.4 hubo una reestructuración fundamental, que toma en cuenta que el núcleo Linux ya está disponible para muchos diferentes arquitecturas de computadoras (diferentes procesadores), y separa elementos independientes de la arquitectura de elementos específico para el CPU o para los buses específicos para un cierto hardware en diferentes directorios.

También el paquete modutils depende de esta distribución y versiones anteriores de este software no encuentran los módulos de núcleos mas recientes.

16.3.3. Otros componentes

En el directorio /boot también se copian los archivos config-<versión> del núcleo, que contiene todos los parametros de compilación del núcleo en cuestión, y el archvio System.map-<versión> que contiene los símbolos exportados.

Los felices que utilizan el sistema Debian pueden contar con que la utilidad "kernel-package" copie estos archivos y los módulos automáticamente en el lugar correcto. "make-kpkg kernel-image" crea un paquete debian (.deb) en el directorio /usr/src, que puede ser "instalado" con el comando "dpkg -i kernel-image-<versión>_Custom.1.00_i386.deb". Este paquete contiene todos los componentes arriba mencionados, los copia en sus lugares correctos y ejecuta lilo para instalar el núclo en el sector de arranque. Si se trata de instalar un núcleo con una versión ya instalado, el programa se rehusa a copiar los módulos y pide reconfirmación. Si se está seguro de querer sobreescribir la versión actual del núcleo de esta versión no se debe insistir, sino primero borrar el directorio con los módulos por completo: rm -rf /lib/modules/<version>, de otra manera se mantienen posiblemente archivos de módulos adicionales en este directorio de los cuales depmod genera mensajes de error de dependencia a la hora de iniciar la computadora.

No es recomendable borrar los módulos del núcleo de instalación. "kernel-package" permite especificar una "sub-versión" para el núcleo a la hora de compilación, con lo cual el nuevo núcleo se distingue aunque sea de la misma versión que un núcleo actual En este caso se instalan los módulos en un directorio diferente. Esto se realiza mediante la opción:

```
make-kpkg --add-to-version <etiqueta> kernel-image
```

donde <etiqueta> puede ser escogido por el administrador del sistema, p.ej. "-actual", o "-1", "-2", etc.

El paquete "kernellab" permite la administración y regeneración automática de muchos versiones y configuraciones de núcleos en una sola computadora, por ejemplo para una máquina "servidor" de núcleos para una red grande y con diferentes computadoras.

16.4. Unidad Zip en puerto paralelo

16.5. SCSI y escaneadores

16.6. Puertos en serie, modems y Plug and Play

16.7. Tarjetas de sonido

16.8. Adaptadores de Red

16.9. Niveles de ejecución y demonios

[rol de demonios en el sistema. Funcionamiento detallado de init y uso de diferentes runlevel - niveles de ejecución.]

17. Programas

[opcional]

17.1. StarOffice

[importante]

[Libre equivalente a MS-Office, Instalación correcta]

17.2. Gnucash

[Programa de contabilidad, opcional]

17.3. Correo Electrónico

[Medio importante]

[Configuración de correo en: varias cuentas, para varios MUA, desde varios servidores externos via Pop3/Multipop, multidrop fetchmail, fetchmail individual por cuenta]

17.4. Fax, Voice

[Fax importante]

[Efax - sencillo, una computadora; hylafax - server, una solución para redes; Voice: no me ha interesado hasta ahora, posiblemente GNUe-telephony=Bayonne]

18. Bonboncitos

Este capítulo da información básica sobre algunos aspectos interesantes e inclusive importantes en el uso de computadoras en general y de Linux en especial. Sin embargo se trata de temas muy amplios o también arbitrariamente seleccionados porque son del interés del autor.

18.1. Respaldos

[¡Muy importante, ya que hay poco conocimiento práctico, poca práctica, y MUCHA necesidad!]

18.2. Fips

[Compagína con Instalación del Sistema Operativo, ya que permite instalar Linux posterior a una instalación existente de Windows sin necesidad de reinstalar Wi. . .]

18.3. TEX/LATEX, dvi y LYX

[Lo mejor en Procesamiento de Palabras, ¡really!, este documento ha sido elaborado con eso y en una velocidad y calidad incomparable con Wor. . . no StarWrite]

18.4. The Gimp

[El sustituto para Photoshop]

18.5. Linuxdoc

[Mhmmm. . . Cómo sacarle mejor provecho a un texto. Generar de una sola fuente una multitud de formatos y presentaciones: Web, TEX, rtf, ASCII, etc. Podría extender también sobre Literate Programming, y Estilo de documentación técnica en Informática (texinfo!!)]

18.6. ps-tools

[¿y también Display Postscript?]

Parte IV.

Anexos

A. Glosario

El byte es la segunda unidad básica para medir volúmen de información, es la unidad más utilizada. La unidad básica es la mínima cantidad de información 0 o 1, es el Bit. 8 Bit se agrupan para formar un Byte (256 posibles entidades de información). 10 Bit expresan 1024 posibilidades, 1024 Byte son 1 KByte. 1K = 1024, 1k = 1000. El megabyte no tiene definición concisa, hay quienes la calculan como

1 M = KxK = 1024x1024 = 1048576

1 M = kxK = 1000x1024 = 1024000

La misma situación ocurre con Gigabyte, GByte.

La unidad traditional de medida en medios para grabar archivos es un bloque = 512 Byte = 1/2 KByte. "du" traditionalmente reporta bloques, en sistemas GNU reporta KBytes. Con la opción -b muestra bytes.

\t tabulador

Chipset Ciertas funciones de Hardware, como la generación de la señal video para el monitor, o el control de los buses de expansión ISA y PCI y el controlador de disco duro que tradicionalmente se realizaron mediante circuitos electronicos individuales son integrados en unos pocos circuitos de alta densidad de integración por algunos productores de componentes electronicos. Estos muchas veces no permiten el uso de componentes diversos, sino requieren que todos los componentes sean de la misma fábrica. Estos "juegos" de Chips (= sinónimo para circuito integrado) son los Chipset, que se denominan según el circuito principal y/o la fábrica. La alta complejidad a veces lleva a diseños que exponen fallas, deficiencias, o incompatibilidades.

Hash de dispersión Una secuencia de caracteres (bytes) puede ser interpretado como un número con muuuchas cifras. A través de operaciones matemáticas (divisiones con polinomios residuales y otros monstruosidades) se crea un número (el resto de la división) con una cantidad fija de cifras = cantidad fija de bytes: el hash, a veces también se le llama checksum. El hash debe ser diferente para cada secuencia de caracteres específica, quiere decir, si se cambia una "cifra" en la secuencia, también debe de cambiar el checksum. Una función de dispersión buena solo tiene muy "pocas" secuencias que producen el mismo checksum respectivo. El checksum entonces simboliza la secuencia de caracteres, es como una firma, o un código corto para ella. La secuencia de caracteres puede ser una clave, un paquete de datos en una transmisión de red, un archivo, etc.

expresiones regulares Description^(techo) al comienzo de la línea

$ al final de la línea

\ protector del carácter siguiente

Comandos de ratón

Click apretar rápidamente un botón

Doble-Click apretar dos veces consecutivamente un botón

Jalar apretar – mover – soltar

Drop soltar un botón apretado

Left-Click, Right-Click apretar el botón izquierdo o derecho del ratón respectivamente.

B. Referencia rápida Emacs

Motion

entity to move over	backward	forward
character	C-b	C-f
word	M-b	M-f
line	C-p	C-n
go to line beginning (or end)	C-a	C-e
sentence	M-a	M-e
paragraph	M-{	M-}
page	C-x [	C-x]
sexp	C-M-b	C-M-f
function	C-M-a	C-M-e
go to buffer beginning (or end)	M-<	M->

scroll to next screen	C-v
scroll to previous screen	M-v
scroll left	C-x <
scroll right	C-x >
scroll current line to center of screen	C-u C-l

Killing and Deleting

entity to kill	backward	forward
character (delete, not kill)	DEL	C-d
word	M-DEL	M-d
line (to end of)	M-0 C-k	C-k
sentence	C-x DEL	M-k
sexp	M-- C-M-k	C-M-k

kill region	C-w
copy region to kill ring	M-w
kill through next occurrence of *char*	M-z *char*
yank back last thing killed	C-y
replace last yank with previous kill	M-y

Marking

set mark here	C-@ or C-SPC
exchange point and mark	C-x C-x
set mark *arg* words away	M-@
mark paragraph	M-h
mark page	C-x C-p
mark sexp	C-M-@
mark function	C-M-h
mark entire buffer	C-x h

Query Replace

interactively replace a text string	M-%
using regular expressions	M-x query-replace-regexp

Valid responses in query-replace mode are

replace this one, go on to next	SPC
replace this one, don't move	,
skip to next without replacing	DEL
replace all remaining matches	!
back up to the previous match	^
exit query-replace	RET
enter recursive edit (C-M-c to exit)	C-r

2

GNU Emacs Reference Card

(for version 20)

Starting Emacs

To enter GNU Emacs 20, just type its name: `emacs`

To read in a file to edit, see Files, below.

Leaving Emacs

suspend Emacs (or iconify it under X)	`C-z`
exit Emacs permanently	`C-x C-c`

Files

read a file into Emacs	`C-x C-f`
save a file back to disk	`C-x C-s`
save **all** files	`C-x s`
insert contents of another file into this buffer	`C-x i`
replace this file with the file you really want	`C-x C-v`
write buffer to a specified file	`C-x C-w`
version control checkin/checkout	`C-x C-q`

Getting Help

The help system is simple. Type `C-h` (or `F1`) and follow the directions. If you are a first-time user, type `C-h t` for a **tutorial**.

remove help window	`C-x 1`
scroll help window	`C-M-v`
apropos: show commands matching a string	`C-h a`
show the function a key runs	`C-h c`
describe a function	`C-h f`
get mode-specific information	`C-h m`

Error Recovery

abort partially typed or executing command	`C-g`
recover a file lost by a system crash	`M-x recover-file`
undo an unwanted change	`C-x u` or `C-_`
restore a buffer to its original contents	`M-x revert-buffer`
redraw garbaged screen	`C-l`

Incremental Search

search forward	`C-s`
search backward	`C-r`
regular expression search	`C-M-s`
reverse regular expression search	`C-M-r`
select previous search string	`M-p`
select next later search string	`M-n`
exit incremental search	`RET`
undo effect of last character	`DEL`
abort current search	`C-g`

Use `C-s` or `C-r` again to repeat the search in either direction. If Emacs is still searching, `C-g` cancels only the part not done.

Multiple Windows

When two commands are shown, the second is for "other frame."

delete all other windows	C-x 1	
split window, above and below	C-x 2	C-x 5 2
delete this window	C-x 0	C-x 5 0
split window, side by side	C-x 3	
scroll other window	C-M-v	
switch cursor to another window	C-x o	C-x 5 o
select buffer in other window	C-x 4 b	C-x 5 b
display buffer in other window	C-x 4 C-o	C-x 5 C-o
find file in other window	C-x 4 f	C-x 5 f
find file read-only in other window	C-x 4 r	C-x 5 r
run Dired in other window	C-x 4 d	C-x 5 d
find tag in other window	C-x 4 .	C-x 5 .
grow window taller	C-x ^	
shrink window narrower	C-x {	
grow window wider	C-x }	

Formatting

indent current line (mode-dependent)	TAB
indent region (mode-dependent)	C-M-\
indent sexp (mode-dependent)	C-M-q
indent region rigidly *arg* columns	C-x TAB
insert newline after point	C-o
move rest of line vertically down	C-M-o
delete blank lines around point	C-x C-o
join line with previous (with arg, next)	M-^
delete all white space around point	M-\
put exactly one space at point	M-SPC
fill paragraph	M-q
set fill column	C-x f
set prefix each line starts with	C-x .
set face	M-g

Case Change

uppercase word	M-u
lowercase word	M-l
capitalize word	M-c
uppercase region	C-x C-u
lowercase region	C-x C-l

The Minibuffer

The following keys are defined in the minibuffer.

complete as much as possible	TAB
complete up to one word	SPC
complete and execute	RET
show possible completions	?
fetch previous minibuffer input	M-p
fetch later minibuffer input or default	M-n
regexp search backward through history	M-r
regexp search forward through history	M-s
abort command	C-g

Type C-x ESC ESC to edit and repeat the last command that used the minibuffer. Type F10 to activate the menu bar using the minibuffer.

3

C. Referencia rápida vi

Este archivo es un resumen y una traducción del documento /usr/doc/nvi/vi.beginner

[next] [prev] [prev-tail] [tail] [up]

Apéndice C
Referencia rápida vi

Este archivo es un resumen y una traducción del documento /usr/doc/nvi/vi.beginner

Acción	Tecla	Acción	Tecla
entrar en vi desde el shell	vi filename	juntar dos líneas	J
salir y guardar	ZZ o :wq<Enter>	encontrar cadena en el archivo:	
salir y descartar	:q!<Enter>	busquando adelante	/ ...string... /^M
salir (no hubo cambios)	:q^M	busquando hacia atrás	? ...string... ?^M
guardar sin salir	:w^M	repetir la última búsqueda	n
guardar con otro nombre	:w nombre^M	repetir la última búsqueda pero en dirección opuesta	N
mover el cursor:		buscar caracter en línea	
derecha	l	busquando adelante	fc
izquierda	h o C-H	busquando hacia atrás	Fc
abajo	j o C-N	repetir la última búsqueda de caracter	
arriba	k o C-P	sustituciones	
insertar texto:		sustituir un caracter con el caracter x	
ante cursor	i	sustituir un caracter con un texto	s ...text...
comienzo de la línea		sustituir n caracteres con un texto	
después del cursor (añadir)	a	sustituir caracteres uno por uno con texto	R ...text...
al final de la línea		deshacer	
después de la línea actual	o	deshacer todos cambios en la línea actual	U
ante la línea actual	O	deshacer el último cambio	
borrar el caracter:			
debajo del cursor	x		
a la izquierda del cursor	X		
borrar n caracteres	#x o #X		
movimiento ámplio:		movimiento ámplio :	
desplazar línea superior de la pantalla	^Y	adelantar una "pantalla"	^F
desplazar línea inferior de la pantalla	^E	retroceder una "pantalla"	^B
subir media pantalla	^U	adelantar una línea	
bajar media pantalla	^D	retroceder una línea	-
mover cursor a la línea superior	H	al inicio de la línea	0
mover cursor a la línea inferior	L	al final de la línea	$
mover cursor a la línea en medio	M	adelantar una palabra	w
marcas:		adelantar una palabra, ignorando puntuación	W

marcar la posición actual, nombrarla x	mx	retroceder al final de la palabra anterior	e
mover cursar a la línea marcada x	☐x	retroceder palabra ignorando puntuacioin	E
mover cursor al caracter marcado con x	☐x	retroceder una palabra	b
mover al comienzo del archivo	1G	retroceder una palabra ignorando puntuación	B
mover al final del archivo	G	regresar a la última línea modificada	☐
mover a la línea 23 del archivo	23G		

redibujar pantalla:

centrar la línea del cursor	z^M
superior de la pantalla	z.
en medio de la pantalla	\-
inferior de la pantalla	z-

Caracteres especiales en cadenas de búsqueda: tienen que ser escapado con \ para usarlos directamente	Caracter
comienzo de la línea	^
final de la línea	$ - (símbolo dolar)
cualquier caracter	. - (punto)
el caracter de escape mismo	\ - (fleca)
para encontrar cadenas según comodínes	[
ditto	]
ditto	* - (asterisco)

[next] [prev] [prev-tail] [front] [up]

Caracteres especiales en cadenas de bu’squeda:tienen que ser escapado con \ para usarlos directamente Caracter comienzo de la li'nea ^final de la li'nea $ - (si'mbolo dolar) cualquier caracter . - (punto)el caracter de escape mismo \ - (fleca) para encontrar cadenas segu'n comodi'nes [ditto] ditto * - (asterisco)

D. Estándard de jerarquía de sistema de archivo

```
/                      directorio raíz
 `--bin                comandos (binarios) esenciales de (todos) usuarios
 `--boot               archivos estáticos del boot loader
 `--dev                archivos de dispositivos
 `--etc                configuración específica del host
 |  |
 |  `--X11             configuración para el sistema X-Windows
 |  `--opt             configuración para /opt
 `--home               directorios personales de usuarios (opcional)
 `--lib                bibliotecas compartidas esenciales
 |  |
 |  `--modules         módulos cargables del núcleo
 `--mnt                punto de montaje temporal
 `--opt                paquetes software adicionales
 `--proc               sistema virtual para información de núcleo y procesos
 `--root               directorio "home" para super-usuario
 `--sbin               binarios de sistema esenciales (para el arranque)
 `--tmp                archivos temporales
 `--usr
 |  |
 |  `--X11R6           X-Windows, Versión 11, Revisión 6
 |  |  |
 |  |  `--bin          binarios de X-Windows
 |  |  `--lib/X11      bibliotecas específicas de X-Windows
 |  |  `--include/X11  archivos de inclusión de X-Windows
 |  `--X386            X-Windows, Ver. 11, Rev. 5 plataformas x86
 |  `--bin             la mayoría de los comandos de usuario
 |  |  |
 |  |  `--mh           commandos para el sistema de correo MH
 |  |  `--X11          enlace simbólico a /usr/X11R6/bin
 |  `--games
 |  `--include         archivos de inclusión estándard
 |  |  |
 |  |  `--X11          enlaces simbólico a /usr/X11R6/include/X11
 |  |  `--bsd          archivos de compatibilidad con BSD (en caso necesario)
 |  |  `--g++          archivos de inclusión de GNU C++
 |  `--lib             bibliotecas para programación y paquetes
 |  `--local
 |  |  |
 |  |  `--bin          binarios locales
 |  |  `--games        binarios de juegos locales
 |  |  `--include      archivos de inclusión locales
 |  |  `--lib          bibliotecas locales
 |  |  `--sbin         binarios de sistema locales
 |  |  `--share        jerarquía independiente de arquitectura local
 |  |  `--src          código fuente local
 |  `--sbin            binarios de sistema estándard no esenciales
 |  `--share           datos independientes de la arquitectura (plataforma)
 |  |  |
 |  |  `--dict         listas de palabras
 |  |  `--doc          documentación diversa
 |  |  `--games        datos estáticos para juegos (/usr/games)
 |  |  `--info         directorio primario para el sistema GNU Info
 |  |  `--locale       información de localización
 |  |  `--man          manuales (unix) en línea
 |  |  |  |
 |  |  |  `--man1      programas de usuario
 |  |  |  `--man2      routinas del sistema
 |  |  |  `--man3      routinas de bibliotecas
 |  |  |  `--man4      archivos especiales (de configuración etc.)
 |  |  |  `--man5      formatos de archivos
 |  |  |  `--man6      juegos
 |  |  |  `--man7      diversos
 |  |  |  `--man8      administración del sistema
 |  |  `--nls          soporte de lenguaje nativo
 |  |  `--misc         datos independientes de arquitectura diversos
 |  |  `--terminfo     base de datos de terminfo
```

```
|   |   `--tmac         macros troff no distribuidos con groff
|   |   `--zoneinfo     información y configuración de zonas de tiempo
|   `--src              código fuente
|       |
|       `--linux        código fuente del núcleo Linux
`--var                  datos variables
    |
    `--account          bitácoras de cuentas (opcional)
    `--cache            datos cache de aplicaciones
    |   |
    |   `--fonts        fuentes generados localmente
    |   `--man          páginas man formateados localmente (opcional)
    |   `--www          datos cache www o datos proxy
    |   `--<paquete>    datos cache de un paquete específico
    `--crash            vuelcos de fallas de sistema (opcional)
    `--games            datos variables de juegos
    `--lib              información de estado variable
    |   |
    |   `--<editor>     Estado y archivos respaldo de editores
    |   `--misc         datos diversos de estado
    |   `--xdm          datos variables de administrador de pantalla X
    |   `--<pkgtool>    archivos soporte de empaquetado
    |   `--<paquete>    datos estáticos para paquetes y subsistemas
    `--lock             archivos de enllavado
    `--log              archivos y directorios bitácoras
    `--mail             apartados de correo de usuarios
    `--opt              datos variables para /opt
    `--run              archivos variables de corrida de procesos
    `--spool            datos cola de aplicaciones
    |   |
    |   `--lpd          directorio cola para impresión
    |   |   |
    |   |   `<impresora> colas de impresión para impresora específica (opcional)
    |   `--mqueue       cola de correo saliente
    |   `--news         cola de News (Noticias)
    |   `--rwho         archivos rwhod
    |   `--smail        directorios cola para smail
    |   `--uucp         directorio cola para UUCP
    `--tmp              archivos temporales preservados durante re-inicios
    `--yp               base de datos para Network Information Service
```

E. Visualizadores

Para la gran variedad de tipos y formatos de archivos hay una igual variedad de utilidades de visualización.

Unix no utiliza en primer lugar el concepto de "extensión" del nombre de archivo para distinguir los formatos, sino deduce del contenido del archivo su tipo. Sin embargo se utiliza las extensiones para facilitar a la usuaria humana el reconocimiento y la administración de sus archivos.

La utilidad "`file`" cuenta con una base de datos de "números mágicos" (magic number), que son los primeros tantos caracteres de un archivo y que generalmente se distinguen según el tipo del archivo.

`file` *nombre*

Utiliza esta base de datos para escribir en la salida estándar el tipo del archivo.

Otro mecanismo que está ganando importancia es la norma MIME (Multipurpose Internet Mail Extensions), que prove una nomeclatura estandardizada para los diferentes formatos y permite una asociación entre tipos de archivos y aplicaciones para su manipulación y/o "visualización" respectiva. Esto último es importante, porque permite que en diferents computadoras, con sistemas operativos o capabilidades diferentes de software instalado siempre se utiliza el "visualizador" adecuado para un cierto tipo de archivos según la asociación MIME individual configurada en esta máquina.

Hay visualizadores especializados para solo un cierto tipo de archivos, pero también hay utilidades genéricas que se adaptan automáticamente a una gran variedad de diferentes formatos de archivos, muchas veces sirviendose de los visualizadores especializados. Los administradores de archivos (Filemanager) normalmente cuentan con un visualizador integrado.

El concepto de un visualizador (browser) es hacer visible de manera rápida el contenido de un archivo en la pantalla y eventualmente imprimirlo, pero no incluye la edición del contenido del archivo en cuestión.

El concepto de un paginador (Pager) es orientado a formatos textuales o flujos (streams) de caracteres y es, que el paginador divide el documento en partes que alcanzan verse en una pantalla o imprimirse en una hoja de papel de un cierto tamaño respectivamente.

more es el paginador tradicional de unix. Su nombre se deriva de que presenta una página de texto y después una línea con el texto "more" = más. Si se aprieta cualquier tecla (menos "q") muestra una página "más". Con "q" interrumpe la paginación.

cat es un filtro unix para concatenar varios archivos, pero puede usarse para imprimire un texto (corto) en la pantalla

head/tail

less “more”, pero con todas las sofisticaciones imaginables, en especial, puede navegarse el texto hacia adelante **y** hacia atrás.

vi/emacs y otros editores pueden usarse como visualizadores y tienen una opción de abrir un archivo de solo lectura. Emacs tiene un sinnúmero de filtros para formatear (y manipular) diferentes tipos de archivos no-textuales. En especial mencionamos el modo “hexl” que permite visualizar archivos (binarios) en forma hexadecimal, y el modo sgml, que permite visualizar archivos sgml, html, xml, etc.

mc es un administrador de archivos en modo ascii/gráfico. Su forma de uso es copiado de Norton Commander (mc = midnight comander) pero tiene funcionalidad muy superior a este (enlaces ftp, http, nfs, ...). mc prove visualizadores para varios tipos de archivos y puede configurarse para visualizadores arbitrarios. En un sistema linux decentemente preconfigurado utiliza una vasta gama de visualizadores individuales para casi toda clase de archivos.

lynx es un navegador html ascii, que utiliza mime para visualizar archivos arbitrarios. De esta manera pueden visualizare los archivos en modo texto (p.ej. convertir un texto MS-Word en forma textual), pero también pueden lanzarse aplicaciones gráficas si se utiliza lynx en ambiente X-Windows.

mozilla o netscape realizan la misma tarea en modo gráfico en X-Windows.

gv,

gqview,-TheGimp son visualizadores para archivos gráficos. eog = Eye of the Gnome, eeyes = Electric Eyes. Los últimos dos no solo son visualizadores sino programas potentes para generar y manipular imágenes/archivos gráficos, mientras los primeros tienen facilidades de manejo de cantidades de imágenes, como p. ej. la de crear listas de iconos de archivos en un directorio (thumbnails). Hay visualizadores de gráficos que pueden usar las capacidades gráficas de la tarjeta SVGA, desde una terminal virtual de Linux: zvg

man, info

tkman, ghelp ameno para usar. ghelp es el visualizador de ayuda del Gnome Desktop y puede visualizar archivos man, info, y html.